AD_________________

Award Number: DAMD17-99-1-9154

TITLE: Model and Expansion Based Methods of Detection of Small Masses in Radiographs of Dense Breasts

PRINCIPAL INVESTIGATOR: Andrew F. Laine

CONTRACTING ORGANIZATION: Columbia University
New York, New York 10027

REPORT DATE: June 2001

TYPE OF REPORT: Annual

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

DISTRIBUTION STATEMENT: Approved for Public Release;
Distribution Unlimited

The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
June 2001

3. REPORT TYPE AND DATES COVERED
Annual (1 Jun 00 - 31 May 01)

4. TITLE AND SUBTITLE
Model and Expansion Based Methods of Detection of Small Masses in Radiographs of Dense Breasts

5. FUNDING NUMBERS
DAMD17-99-1-9154

6. AUTHOR(S)
Andrew F. Laine

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Columbia University
New York, New York 10027

E-Mail: **Laine@columbia.edu**

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. Abstract *(Maximum 200 Words)* (abstract should contain no proprietary or confidential information)

This report describes progress made in during the second year of study. Our goal is to detect masses in dense mammograms having a diameter less than 1 cm. The idea of this project is to detect subtle masses by tuning the central frequency and width of a basis function used in an overcomplete expansion. By modeling the shape of a mass through this flexibility we hope to detect small and subtle masses in dense breasts and improve the chances of early detection in screening mammography.

This phase of our study showed that fractional splines functions are a powerful tool for the representation of masses in mammograms and are well matched to the mass detection problem. We showed that by using a continuously varying parameter of order, that accurate approximations of mass shapes could be obtained through overcomplete expansions of a fractional spline wavelet transform. In the context of addressing the problem of finding the best scale, using a library of bases computed by wavelet packets was an efficient method in finding the best scale. Finally, we ported this algorithm using the *Matlab* compiler to allow integration into our Mammography Computer Aided Diagnosis (CAD) workstation for real-time interactive screening of mammography cases during the third year.

14. Subject Terms (keywords previously assigned to proposal abstract or terms which apply to this award)
Detection, Masses, Multiscale Analysis, Wavelets, CAD

15. NUMBER OF PAGES
43

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
Unlimited

Table of contents

1. INTRODUCTION

This report describes progress made in during the second year of study on the project entitled "Model & Expansion Based Methods of Detection of Small Masses in Radiographs of Dense Breasts." Our end goal is to develop technology for computer aided diagnosis that can detect masses in dense mammograms having a diameter less than 1 cm. The basic "idea" of this IDEA category award is to detect subtle masses by tuning the central frequency and width of a basis function. By modeling the shape of a mass through this flexibility (i.e., changing the shape of the "bump") we hope to better detect small and subtle masses in dense breasts and improve the chances of early detection through screening mammography.

This annual report of our second year continues progress made towards accomplishing the goals described in Tasks 1a and 1b, "Feasibility assessment and design of model based method." In addition we have begun work towards building a detector as described in in Task 2b, "Detector analysis and implementation."

During the past year we have focused on an interesting family of functions that are well suited to the problems of mass detection: fractional B-splines are an extension of the B-spline functions. In this report, we study the influence of the spline parameter in the analysis of mammograms and we describe experiments carried with aim of identifying parameters most efficient in characterizing the masses in dense breasts.

Recall that in our previous report, we observed that masses within mammograms are represented low frequency signals.. Thus the analyzing functions we use should to be smooth in terms of shape and can, for instance be first or second derivative functions. The fractional spline family is exactly what we want since it has one continuously varying order parameter. Thus, one can tune the central frequency of a wavelet that will be efficient in the detection of masses of arbitrary size.

2. BODY

It has been shown that the dyadic transform is not sufficient for the case of mass detection [1]. Thus we have constructed a 2D-continuous discrete transform, using the "A trou Algorithm" in which we incorporated the fractional spline wavelet. In addition, we will report on a component of a detector algorithm which uses a dictionary of bases computed from Wavelet Packets. Our goal related to Task 2, was to find an "average" best basis according to minimizing a criterion of energy within a wavelet packet expansion.
In this report, we have processed numerous Regions of Interest (ROI) to allow us to compute this average best basis and extract some specific features for the detection of small masses.

Design and Methods

The 2D fractional B spline Wavelet Transform

The B spline functions

Splines [2] are piecewise polynomials that are smoothly connected together. The joining points of the polynomials are called knots. For a spline of degree *n*, each segment is a polynomial of degree *n*, which suggests that we need (*n*+1) coefficients to describe each piece. However, there is additional smoothness constrain that imposes continuity on the spline and its derivatives up to order (*n*-1) at the knots, so that, effectively, there is one degree of freedom per segment. Here, we will only consider splines with uniform knots and units spacing. The remarkable result is that these functions are uniquely characterized in terms of B-spline expansions:

$$s(x) = \sum_{k\in\mathbb{Z}} c(k)\beta^n(x-k) \qquad (0.1)$$

which involve integer shifts of the B-spline of degree n denoted by β^n ; the parameters are the B-spline coefficients c(k). B-splines, defined below are bell-shaped and constructed from the (n+1) fold convolution of a rectangular pulse β^0 (centered):

$$\beta^0(x) = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ \frac{1}{2}, & |\mathrm{x}| = \frac{1}{2} \\ 0, & \text{otherwise} \end{cases} \qquad (0.2)$$

The fractional B-spline functions [2,3]

We can extend B-Spline to all fractional degrees $\alpha > -\frac{1}{2}$. These splines are constructed using linear combinations of the integer shifts of the power functions x_+^α (one-sided) or $|x|_*^\alpha$ (symmetric):

$$x_+^\alpha = \begin{cases} x^\alpha & x \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

$$|x|_*^\alpha = \begin{cases} \dfrac{|x|^\alpha}{-2\sin\left(\frac{\pi}{2}\alpha\right)}, & \alpha = 2n+1 \\ \dfrac{x^{2n}\log x}{(-1)^{1+n}\pi}, & \alpha = 2n \end{cases}$$

In each case they are α-Holder continuous for $\alpha > 0$. They satisfy most of the properties of the traditional B-splines, as described below. In particular the Riesz basis condition and the 2-scale relation, which make them suitable for the construction of a new families of wavelet bases.

Definition and notation

Euler's gamma function is defined as:

$$\Gamma(u+1) = \int_0^{+\infty} x^n e^{-x} dx \qquad (0.3)$$

In particular, $\Gamma(n+1) = n!$. This suggests the following generalization of the Binomial coefficients

$$\binom{u}{v} := \frac{\Gamma(u+1)}{\Gamma(v+1)\Gamma(u-v+1)} \qquad (0.4)$$

In particular, this definition implies that $\binom{u}{k} = 0$ when k is a strictly negative integer. Moreover, for $u > 0$, we have the well-known binomial theorem: $(1+z)^u = \sum_{k=0}^{+\infty} \binom{u}{k} z^k$.

The differentiation operator can be extended to non-integer exponents rather simply in the Fourier Domain:

$D^{\alpha} f(x) \leftrightarrow (j\omega)^{\alpha} \hat{f}(\omega)$ where $\hat{f}(\omega) = \int f(x) e^{-j\omega x} dx$ denotes the Fourier Transform of $f(x)$ and where $z^{\alpha} = |z|^{\alpha} e^{j\alpha \arg(z)}$ with $j = \sqrt{-1}$ and $\arg(z) \in [-\pi \; \pi]$. This is equivalent to Liouville's definition of fractional derivative operator.

The forward fractional finite difference operator of order α is defined as

$$\Delta_{+}^{\alpha} f(x) = \sum_{k=0}^{+\infty} (-1)^k \binom{\alpha}{k} f(x-k). \qquad (0.5)$$

It is a convolution operator which has a more straightforward interpretation in the Fourier domain

$$\Delta_{+}^{\alpha}(\omega) = (1 - e^{-j\omega})^{\alpha} = \sum_{k=0}^{+\infty} (-1)^k \binom{\alpha}{k} e^{-j\omega k}. \qquad (0.6)$$

The natural building blocks for the fractional splines are the one-sided power functions x_{+}^{α}, which have precisely one singularity of order α (Holder exponent) at the origin. Their Fourier transform is $\Gamma(\alpha+1)/(j\omega)^{\alpha+1}$.

Similar formulae exist for the symmetric functions whose Fourier transform is $\Gamma(\alpha+1)/|j\omega|^{\alpha+1}$. However, in our study, we only consider the one-sided functions.

Fractional B-splines

We consider here the causal fractional B-splines. By analogy with the classical B-spline we define the fractional B-spline by taking the $(\alpha+1)th$ fractional difference of the one-sided power function

$$\beta_{+}^{\alpha}(x) = \frac{\Delta_{+}^{\alpha} x_{+}^{\alpha}}{\Gamma(\alpha+1)} = \frac{1}{\Gamma(\alpha+1)} \sum_{k=0}^{+\infty} (-1)^k \binom{\alpha+1}{k} (x-k)_{+}^{\alpha}. \qquad (0.7)$$

These functions are L_2 for $\alpha > -\frac{1}{2}$. While they seem to be decaying reasonably rapidly, they are not compactly supported unless α is an integer, in which case we recover the classical B-splines. In general, they do not have an axis of symmetry either. The fractional causal B-splines satisfy the convolution property

$$\beta_{+}^{\alpha_1} * \beta_{+}^{\alpha_2} = \beta_{+}^{\alpha_1+\alpha_2+1}. \qquad (0.8)$$

The properties of fractional splines

The Fourier transform of the causal fractional B-splines are given by

$$\hat{\beta}_{+}^{\alpha}(\omega) = \left(\frac{1-e^{-j\omega}}{j\omega} \right)^{\alpha+1}. \qquad (0.9)$$

This equation is indeed compatible with the convolution property.

Fractional derivatives and regularity

One of the primary reasons for the success of polynomial splines in image and signal processing applications is that they can be differentiated very simply by taking finite differences. This property generalizes nicely to the fractional case:

$$D^{\gamma} \beta_{+}^{\alpha}(x) = \Delta_{+}^{\gamma} \beta_{+}^{\alpha-\gamma}(x) \qquad (0.10)$$

(The 2 scale relation is described in the next section).

Fractional spline spaces and Riesz bounds

The mathematical space of fractional splines of degree α with knots at the integers is defined as:

$$S_+^\alpha = \left\{ s(x) = \sum_{k\in\mathbb{Z}} c(k)\beta_+^\alpha(x-k) : c \in l_2 \right\}.$$

Proposition: *For α>-1/2 the fractional B spline* β_+^α *generates a Riesz basis of* S_+^α. *Specifically, one has the following* l_2*-*L_2 *norm equivalence*

$$\forall c \in L_2,\quad A_\alpha \|x\|_{l_2}^2 \le \left\| \sum_{k\in\mathbb{Z}} c(k)\beta^\alpha(x-k) \right\|_{L_2} \le B_\alpha \|x\|_{l_2}^2$$

$$\text{Where}\quad A_\alpha \ge \left(\frac{2}{\pi}\right)^{2\alpha+2} \quad\text{and}\quad B_\alpha \le 1 + 2\zeta(2\alpha+2)\left(\frac{2}{\pi}\right)^{2\alpha+2}$$

This result ensures that the B-spline representation is stable and that fractional spline spaces are well-defined (closed) subspaces of L_2. Finally, the fractional splines of degree α have a fractional order of approximation α+1.

Definition of the fractional spline wavelet

Let us now show explicitly how to use these B-splines to obtain new wavelet families with a continuously varying order parameter. We will concentrate on the case of orthogonal wavelet bases. These are obtained by orthogonalization of the fractional B-splines. The key quantity is the fractional B-spline autocorrelation sequence:

$$a_\varphi(k) := \left\langle \beta_+^\alpha(x), \beta_+^\alpha(x-k) \right\rangle = \beta_*^{2\alpha+1}(k) \qquad (0.11)$$

Where the right hand side expression is a direct consequence of the convolution relations. Moving to the Fourier domain we get

$$A_\varphi(e^{j\omega}) := \sum_{k\in\mathbb{Z}} \left| \hat{\beta}_+^\alpha(\omega + 2\pi k) \right|^2 = \sum_{k\in\mathbb{Z}} \beta_*^{2\alpha+1}(k) e^{-j\omega k} = A_\varphi^\alpha(e^{j\omega}), \; (0.12)$$

since $A_\varphi^\alpha(e^{j\omega}) = \sum_{n\in\mathbb{Z}} e^{-jn\omega} \int \beta_+^\alpha(x)\beta_+^\alpha(x+n)dx.$

To compute $A_\varphi(e^{j\omega})$ we choose to perform an infinite summation in the Fourier domain because of the relatively convenient expression of $\hat{\beta}_+^\alpha(\omega)$. And we express the orthogonal scaling function as

$$\Phi(x) = \sum_{k\in\mathbb{Z}} (a_\varphi)^{-1/2} \beta_+^\alpha(x-k). \qquad (0.13)$$

The corresponding two-scale relation is

$$\Phi(x/2) = \sqrt{2} \sum_{k\in\mathbb{Z}} h_0(k)\Phi(x-k) \qquad (0.14)$$

and the refinement filter is given by

$$H_0(e^{j\omega}) = \sqrt{\frac{A_\varphi^\alpha(e^{j\omega})}{A_\varphi^\alpha(e^{j2\omega})}} \sqrt{2} \left(\frac{1-e^{-j\omega}}{2} \right)^{\alpha+1}. \qquad (0.15)$$

The corresponding orthogonal wavelet filter can then be obtained using Mallat's method:

$$G_0(z) = z.H_0(-z^{-1}). \qquad (0.16)$$

An overcomplete representation for signals: the "A trous Algorithme"[3]

We have shown that the fast dyadic transform was not sufficient in the case of small mass detection [1]. However, it is possible to sample the time-scale space in other ways. The Continuous Discrete Wavelet Transform (CDWT) keeps the dyadic sampling of scale (Discrete) but provides a quasi-continuous sampling in time (Continuous). It is similar to a fast biorthogonal wavelet transform, without downsampling in the time domain. In terms of representation, this transform falls between the large redundancy of the CWT and the strictly non-redundant DWT.

$$cd_x(j,k) = CWT_x(a = 2^j, t = k)$$

$$cd_x(j,2^k) = DWT_x(j,k)$$

Proposition: For any $j \geq 0$,

$$\begin{aligned} a_{j+1}[n] &= a_j * h_j[n] \\ d_{j+1}[n] &= d_j * g_j[n] \end{aligned} \quad \text{and} \quad a_j[n] = \frac{1}{2}\left(a_{j+1} * \tilde{h}_j[n] + d_{j+1} * \tilde{g}_j[n]\right)$$

where a_j and d_j are the approximation and the detail coefficients of the signal

$$h_j = \uparrow 2^{j-1}[h_1] \text{ and } g_j = \uparrow 2^{j-1}[g_1] \qquad (0.17)$$

h_1 and g_1 are respectively the low-pass and high pass filters linked to the wavelet.

:

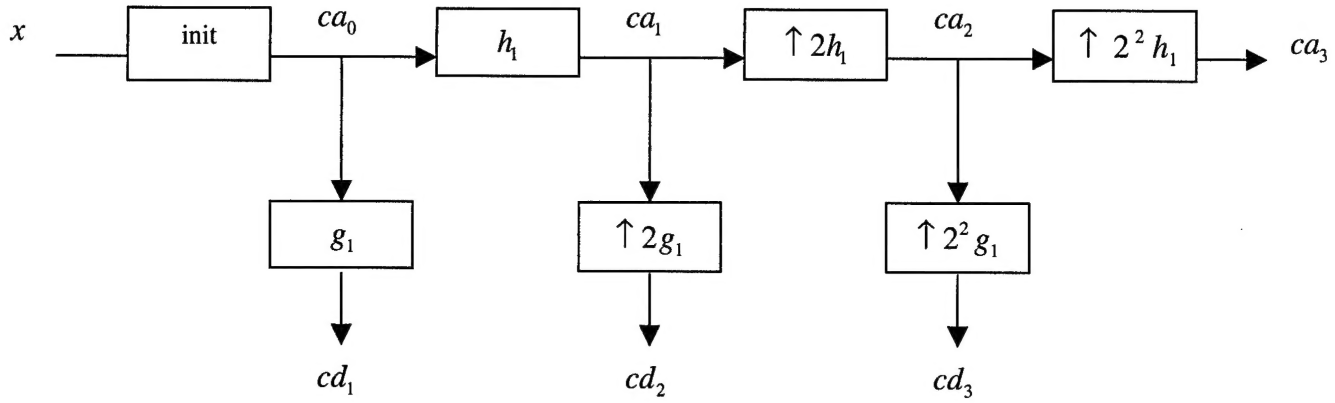


Figure 1. Overview of processing to compute the 'a trou' algorithm.

The 2D Continuous Discrete Fractional Spline Wavelet Transform

Implementation

As described above, we used the "a trou Algorithm" to perform this transform. We appled the low-pass and high-pass filters respectively on the rows and the columns as illustrated in Figure 1. We worked in the Fourier domain: ca_0 is the Fourier Transform of the image, the initialization is therefore ensured by the Fourier operation. After the filtering operations, an inverse Fourier transform is performed to obtain the coefficients in the spatial domain.

Filters h_1 and g_1 are respectively linked to H_0 and G_0 of (1.15) and (1.16) with $G_0(e^{j\omega}) = e^{-j\omega}.H_0(-e^{-j\omega})$. In order to have normalized Quadratic Mirror Filters (QMF), we consider $\frac{H_0}{\sqrt{2}} = h_1$ and $\frac{G_0}{\sqrt{2}} = g_1$.

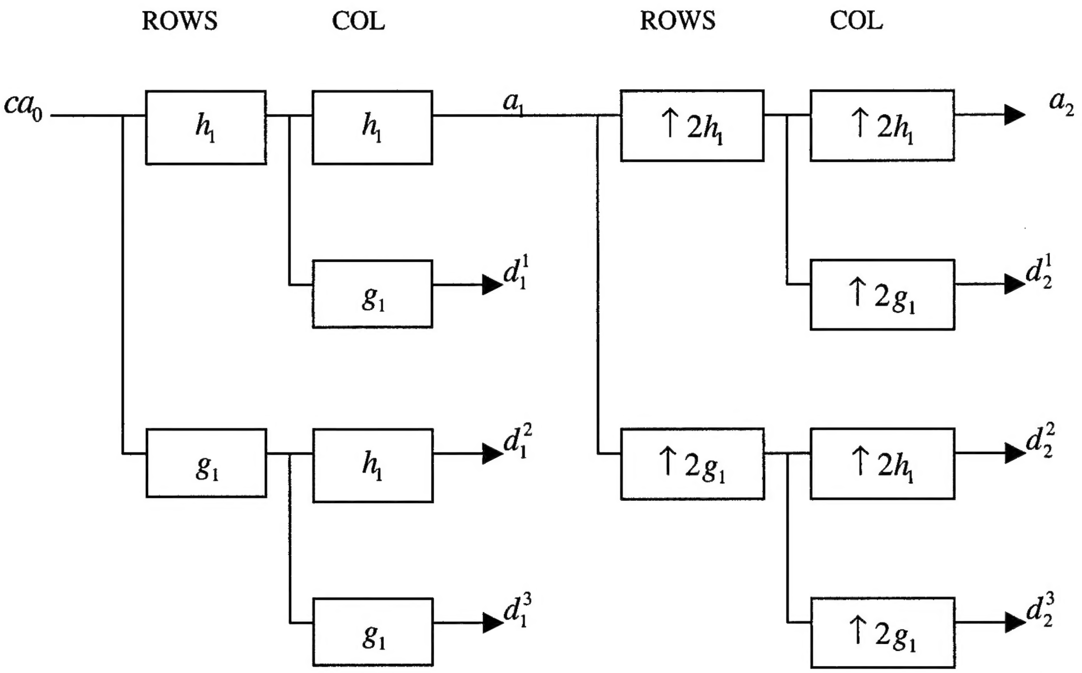


Figure 2: Scheme of the overcomplete algorithm, a_i : Approximation in the wavelet basis, d_i^1 : Horizontal detail in the wavelet basis, d_i^2 : Vertical detail in the wavelet basis, d_i^3 Diagonal detail in the wavelet basis.

Practical considerations:

For the computation [0.6] of the Fourier transforms of the fractional B-splines filters we need to evaluate the autocorrelation filter at $\nu = \frac{k}{n}$, for $k = 0 \ldots N-1$. For this we use the Poisson-equivalent expression of (6) i.e., $\sum_{k\in\mathbb{Z}} \left|\hat\beta_+^\alpha(\omega + 2\pi k)\right|^2$. More precisely, since we must limit this summation to a finite number of terms, we evaluate A_φ using the following asymptotic equivalent:

$$A_\varphi(e^{j2\pi\nu}) \approx \sum_{|n|\le N} \left|\hat\beta_+^\alpha\left(2\pi\left(\nu+n\right)\right)\right|^2 + \left|\frac{\sin\left(\pi\nu\right)}{\pi}\right|^{2\alpha+2}\left[\frac{\frac{2}{2\alpha+1}}{N^{2\alpha+1}} - \frac{1}{N^{2\alpha+2}} + \frac{(\alpha+1)\left(\frac{1}{3}+2\nu^2\right)}{N^{2\alpha+3}} - \frac{(\alpha+1)(2\alpha+3)\nu^2}{N^{2\alpha+4}}\right]. \tag{0.18}$$

An accurate analysis of this expression shows indeed that the remainder difference between the lhs and the rhs is $O\left(\frac{1}{N^{2\alpha+5}}\right)$, which ensures an accuracy that is larger than 200 dB with N=100 computed terms, and for all values of $\alpha > -0.5$. (The code implementing this is provided in the Appendix of this report.)

Moreover, we use biased coefficients to ensure positive coefficients, and display a gray scale.

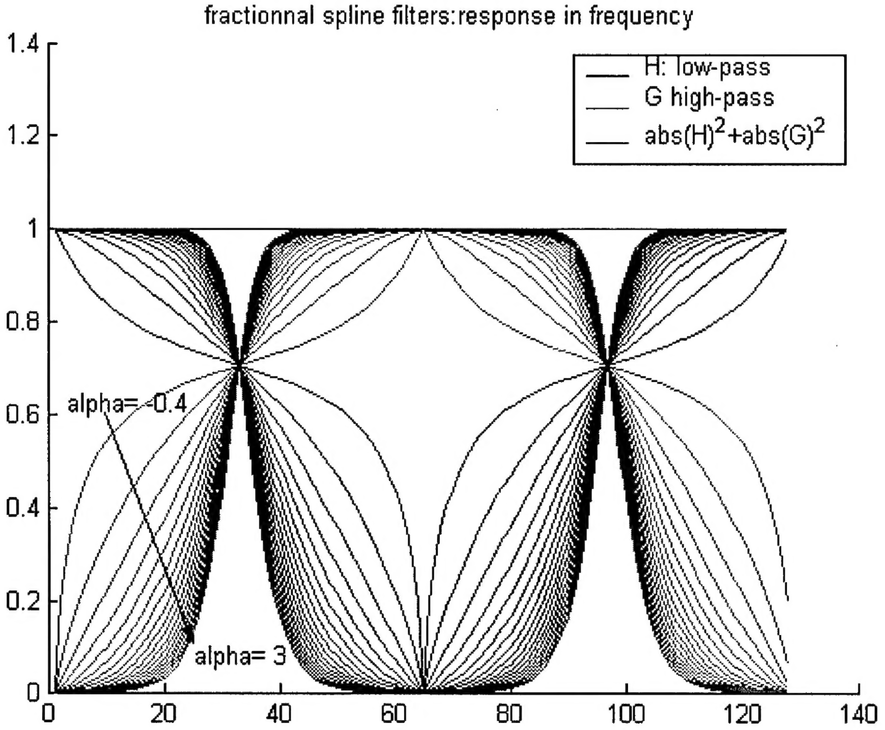


Figure 3: Filter design as a function of alpha, for a 128 points signal.

The filters are a quadratic mirror filter, which are very desirable for signal processing - it allows us to use existing tools that can be applied to such filters. This is also the case for the wavelet packet representation described below, next.

Best basis Search for Detection

The 1D wavelet packet representation

Wavelet packets generalize the link between multiresolution approximations and wavelets. Previously, we noted [5] that a space V_j of a multiresolution approximation is decomposed in a lower resolution space V_{j+1} and a detail space W_{j+1} by dividing the orthogonal basis of V_j into two new orthogonal basis, one for V_{j+1} and the other for W_{j+1}.

But here, instead of dividing only the approximation space V_j to construct the detail space W_{j+1} and a new approximation space V_{j+1}, we want also to divide W_j to derive new bases! The following theorem allow us to perform such divisions:

Theorem : (Coifman, Meyer, Wickerhauser)

Let $\left\{\theta_j(t-2^j n)\right\}_{n\in\mathbb{Z}}$ be an orthonormal basis of a space U_j. Let h and g be a pair of conjugate mirror filters. Define:

$\theta^0_{j+1}(t)=\sum_{n=-\infty}^{n=+\infty} h[n]\theta_j(t-2^j n)$ and $\theta^1_{j+1}(t)=\sum_{n=-\infty}^{n=+\infty} g[n]\theta_j(t-2^j n)$, the family $\left\{\theta^0_{j+1}(t-2^{j+1}n),\theta^1_{j+1}(t-2^{j+1}n)\right\}_{n\in\mathbb{Z}}$ is an orthonormal basis of U_j.

Suppose our signal is approximated at the scale 2^L. The corresponding approximation space is V_L and admits the following orthogonal basis: $\left\{\phi_L(t-2^L n)\right\}_{n\in\mathbb{Z}}$.

We can easily represent the recursive splitting of vector spaces in a binary tree - V_L is the root of the tree, and to each node (*j*,*p*) we associate a detail space W_j^p that admits the following orthonormal basis: $\left\{\psi_j^p(t-2^j n)\right\}_{n\in\mathbb{Z}}$. At the root, we have: $W_L^0 = V_L$. For each node, *j*-L is its depth in the tree and *p* is the number of nodes that are on its left at the same depth. Suppose now that we have already constructed W_j^p. The construction of its children W_j^p and W_j^p is given by the splitting relation of the last theorem:

$$\begin{cases} \psi_{j+1}^{2p}(t) = \sum\limits_{n=-\infty}^{n=+\infty} h[n]\psi_j^p(t-2^j n) \\ \psi_{j+1}^{2p+1}(t) = \sum\limits_{n=-\infty}^{n=+\infty} g[n]\psi_j^p(t-2^j n) \end{cases} . \qquad (0.19)$$

And since $\left\{\psi_{j+1}^{2p}(t-2^{j+1}n), \psi_{j+1}^{2p+1}(t-2^{j+1}n)\right\}_{n\in\mathbb{Z}}$ is and orthonormal basis of W_j^p we have:
$W_j^p = W_{j+1}^{2p} \oplus W_{j+1}^{2p+1}$.

Following this recipe, we constructed a binary tree of wavelet packet bases. We may choose to divide each node further (or not) so that we obtain a binary tree where each node has either 0 or 2 children. A tree is called *admissible* if at the end of the exspansion: $V_L = \oplus_1^I W_{j_i}^{p_i}$ where $\left\{j_i, p_i\right\}_{1\le i\le I}$ are the leaves of our tree.

The union of the corresponding wavelet packet bases $\left\{\psi_{j_i}^{p_i}(t-2^{j_i}n)\right\}_{\substack{n\in\mathbb{Z}\\ 1\le i<I}}$ thus defines an orthogonal basis of $W_L^0 = V_L$: it is called *a wavelet packet basis.*

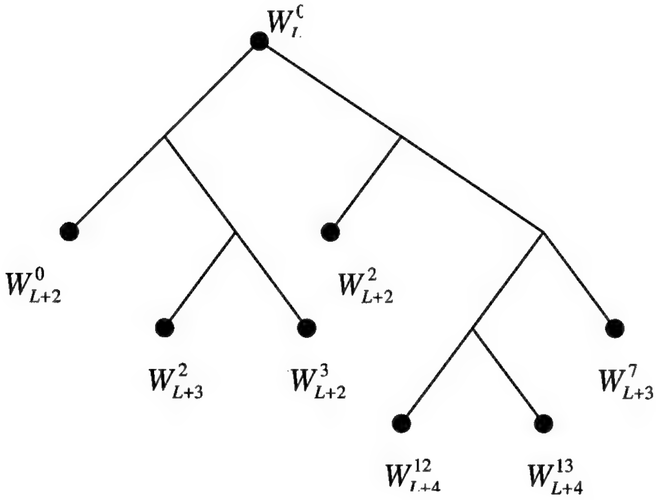


Figure 4: Example of an admissible tree computed from a wavelet packet expansion.

Best basis search

We implemented the wavelet packet decomposition in two dimensions. We can imagine Figure 4 above with 4 children per node (one approximation node and three details nodes for each expansion). The search for a best basis uses an additive criterion described below.

The criterion:

As an early choice in our experiments, we selected the following Entropy additive criteria:

$$Ent = -\sum_{k=1}^{n^2} p_k \log_2(p_k) \text{ with } p_k = \frac{x_k^{\ 2}}{\left\|x_k\right\|^2}, x_k = \left(x_{i,j}\right)_{(i,j)\in[\![1,n]\!]^2} \qquad (0.20)$$

where x is the image matrix.

Algorithm in 2D:

Initialization: $A_{J\max}^{p,k} = \psi_{J\max}^{p,k}, (p,k) \in [\![0,...,2^{J\max}-1]\!]^2$

For $j = J_{\max} - 1,\ 0$:

$$A_j^{p,k} = \begin{cases} \psi_j^{p,k} & \text{if } Ent(\psi_j^{p,k}) \le Ent\left(A_{j+1}^{2p,2k}\right) + Ent\left(A_{j+1}^{2p+1,2k}\right) + Ent\left(A_{j+1}^{2p,2k+1}\right) + Ent\left(A_{j+1}^{2p+1,2k+1}\right) \\ A_{j+1}^{2p,2k} \oplus A_{j+1}^{2p+1,2k} \oplus A_{j+1}^{2p,2k+1} \oplus A_{j+1}^{2p+1,2k+1} & \text{otherwise} \end{cases}$$

$p = 0,...,2^j - 1$

The selected leaves of the tree are the best basis.

The criteria is an entropy-based metric that has been used in application of image compression: Its purpose is to concentrate in a few coefficients most of the information of an image. In our case, the highest value coefficients can describe the mass if the wavelets are well adapted for detection (Please see discussion section below).

Implementation notes

We needed to adapt the algorithm of the decomposition of wavelet packets for the end goal of detection. The 2D discrete decomposition already existed in a **Wavelab** library. We have modified this available code for our purpose, as described in the Appendix of this report (cf *best basis directory*). In particular, as we did not want to downsample coefficients we needed to preserve normalization of the entropy when we compared the entropies of the father to the children nodes in the tree. Thus we divided the children's entropy by a factor of 4.
We also stored the different coefficients of each best basis for reconstruction and visualization purposes.
Thus, we forced the first node to the value 1 in order to allow the program *sb_Plot2dpartition.m* to decompose the nodes. This code also provided in the Appendix, under the heading of Best basis directory.

Results

Computation of the 2D Wavelet transform

We computed along the dyadic scales, the CDWT2D for different parameter values of α (please see the subroutine "*simu_alpha_scale.m*" in the Appendix for details). The results in terms of detection efficiency, are discussed in the next section. .First, we choose a range of α values that made sense in terms of detecting masses less than 1cm in diameter.
First we computed an expansion for a phantom image of a mass (Figure 6) with a large amount of Gaussian noise added (Figure 7). This experimental phantom was used to validate (debug) our algorithm and confirm function.

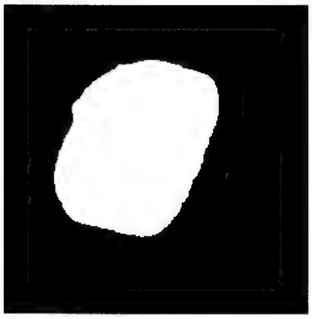


Figure 6: mass model

Gaussian noise
V=4

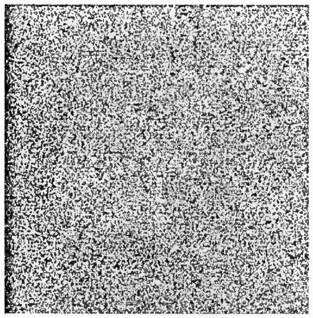


Figure 7: noisy mass

This page intentionally blank.

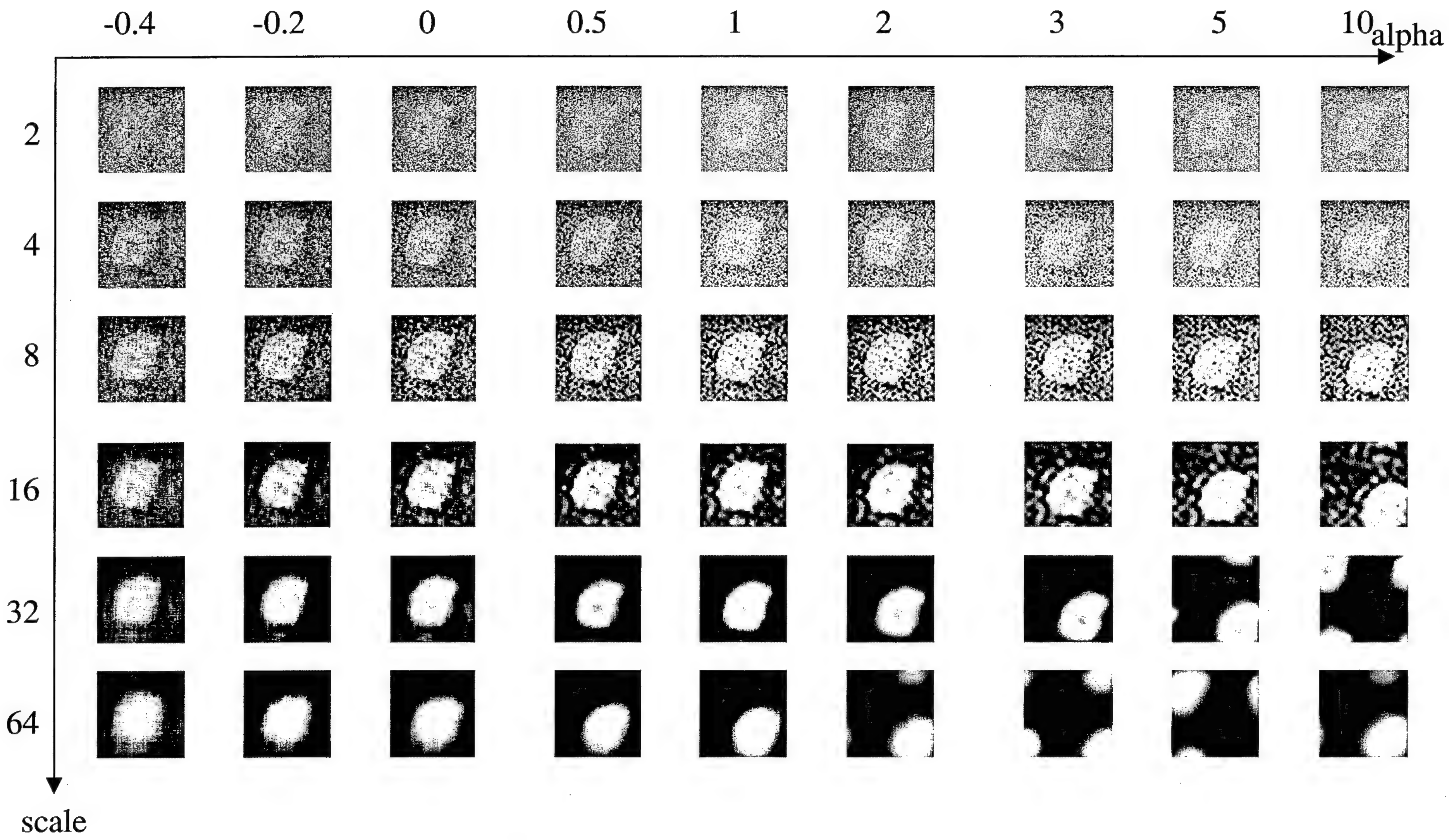


Figure 9: Expansion of approximation coefficients of the CDWT2D for a phantom containing a noisy mass.

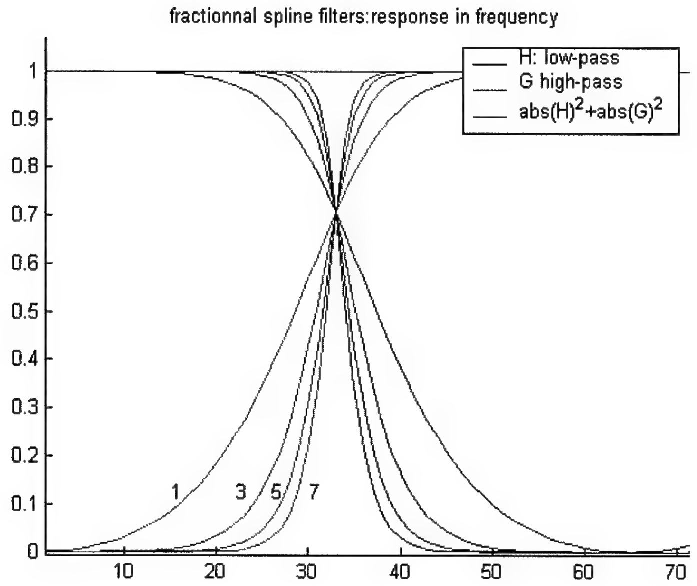


Figure 10: Filter frequency response for corresponding values of selected α.

We tested the transform for the following α values shown in Figures 10 and 11. This range in alpha values was selected because the filters had smooth slopes, desirable for matching the shape of masses.

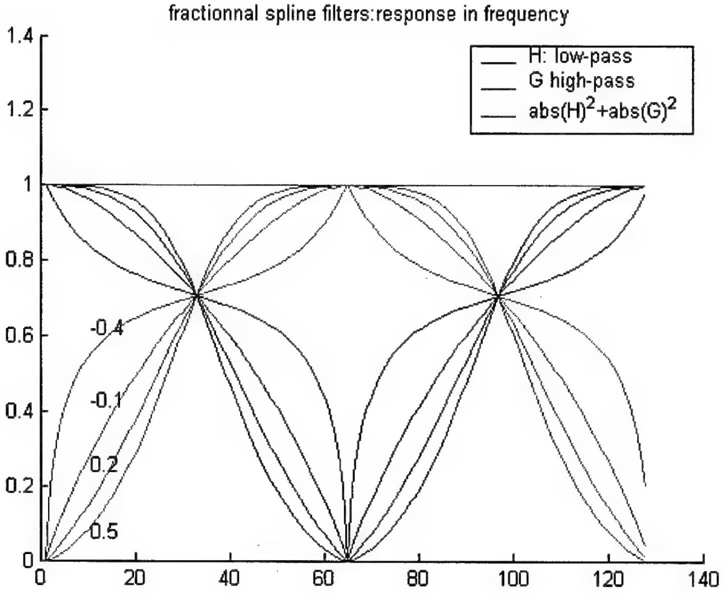


Figure 12

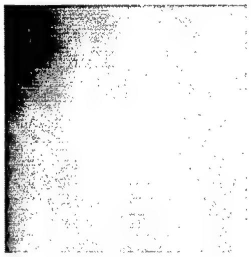
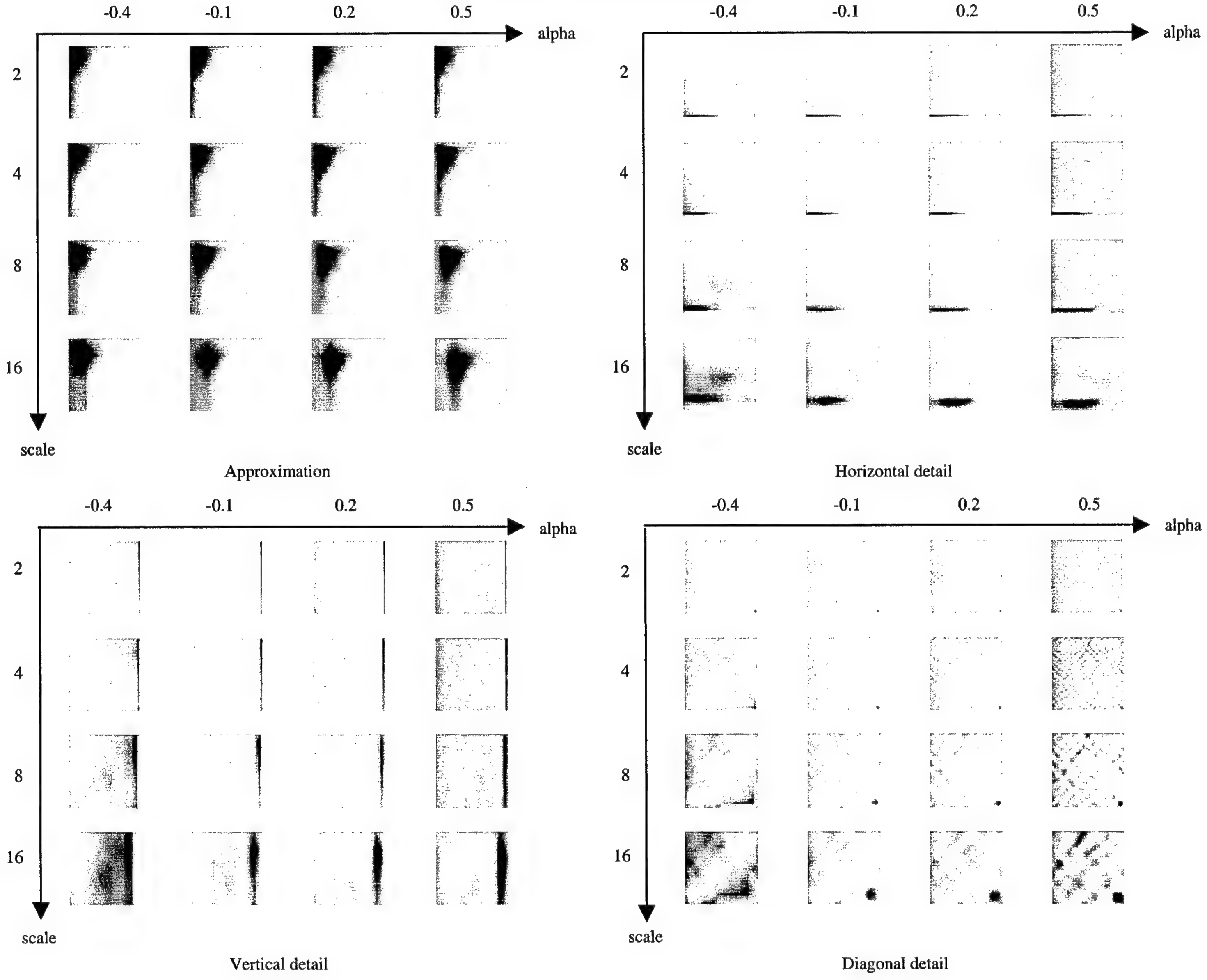


Figure 13: Sample ROI of a barely visible mass.



Approximation

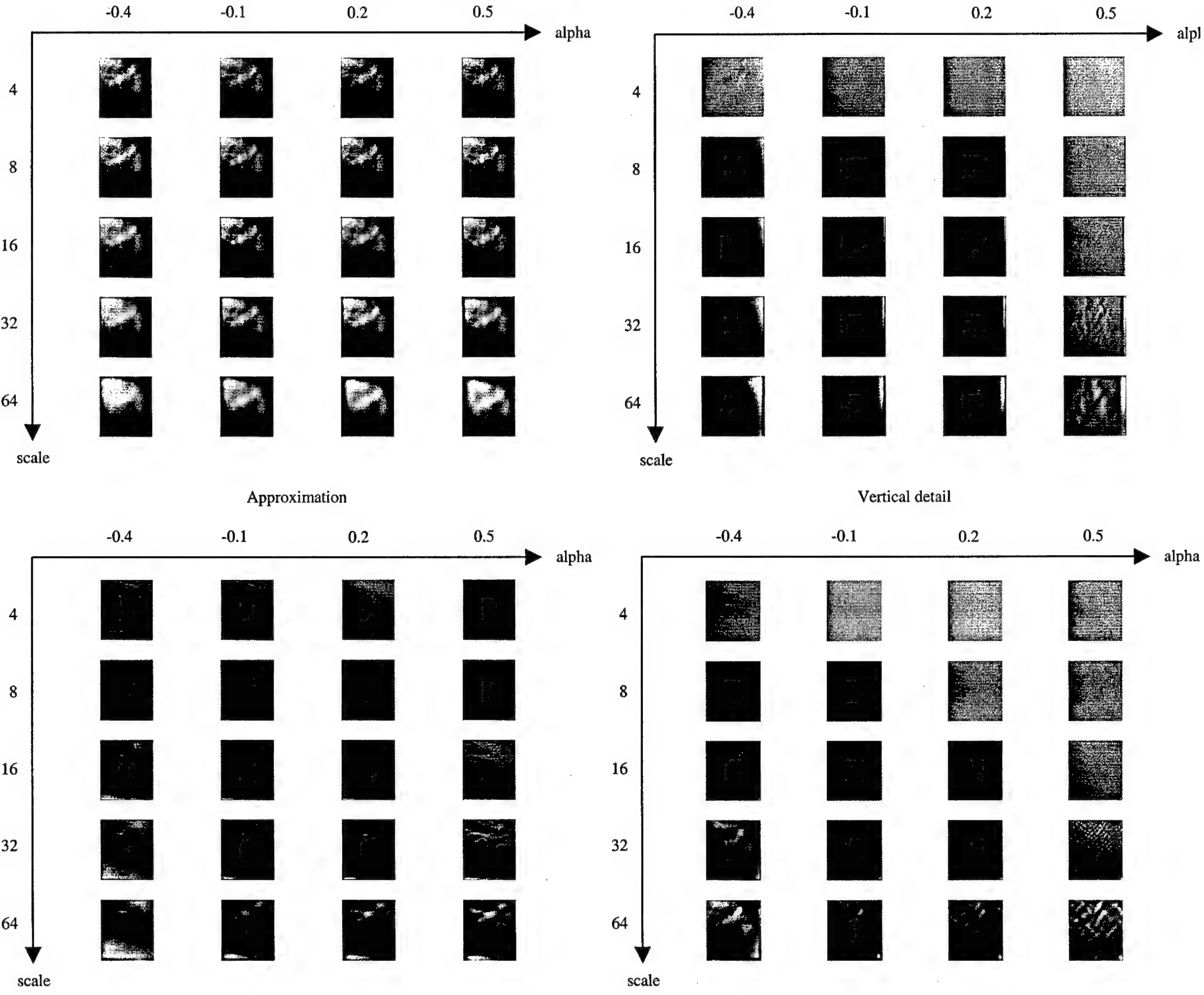
Horizontal detail

Vertical detail

Diagonal detail

Figure 14: CDWT2D of Figure 13, an image matrix size of 64x64.

Figure 15: ROI Case 02, poor exposure, mass located close to skin surface.



Approximation

Vertical detail

Horizontal detail

Diagonal detail

Figure 16: CDWT2D expansion of case 02.

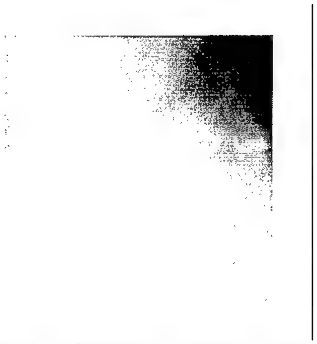


Figure 17:ROI case 04, a spicular mass.

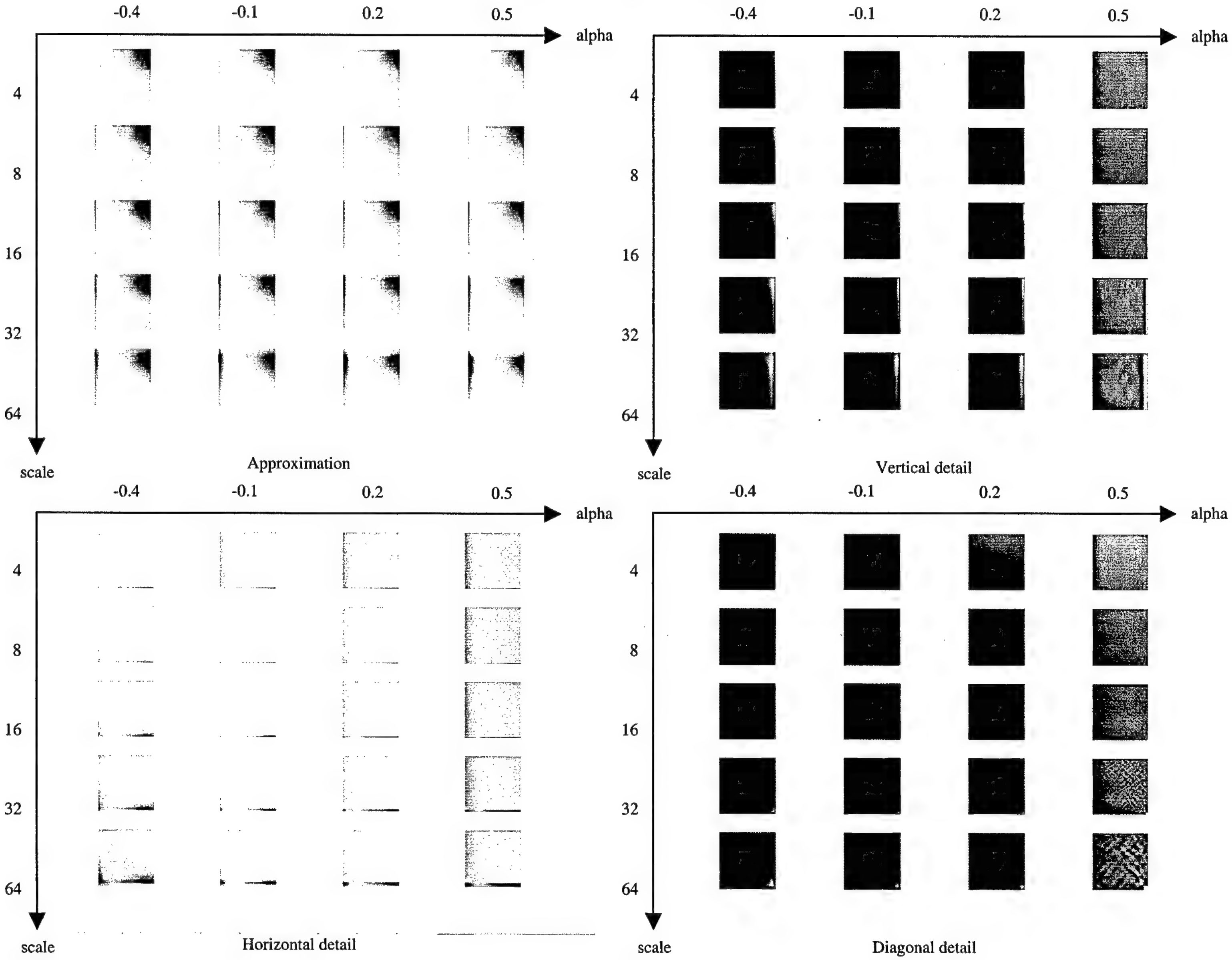


Figure 18: CDWT2D expansion of case 04.

Best basis search

We applied the best basis search strategy described above, on images extracted from the database of mammograms available on the University of South Florida website http://marathon.csee.usf.edu/Mammography/Database.html. We wrote a routine that allows us to choose regions of interest within these mammograms (512x512 pixels). Each case contained a cancerous masses and their features are detailed in the following table:

Cases	Properties
02,04,05,06,08,14,17, 18,21,25,26,27,28,29,30	Type: cancer Density: 4 Lesion type: mass Mass options: Irregular and lobular

The returned results are trees that correspond to the best basis. For practical visualization and practicality, these results are displayed in terms of 2D trees that are computed as if the images were not down-sampled.

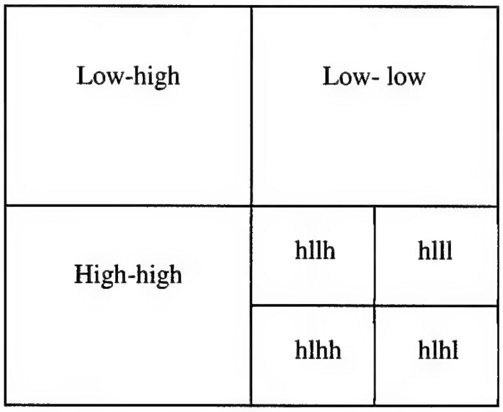


Figure 19. Display format of filtered coefficients using a wavelet packet expansion.

All processed images were 512x512 matrix size and the computation was time-consuming in terms of processing. We therefore decide to limit our expansion to a depth of 4, which corresponds to a scale 2^4=32. We also used the filters shown in Figure 20 below for computation.

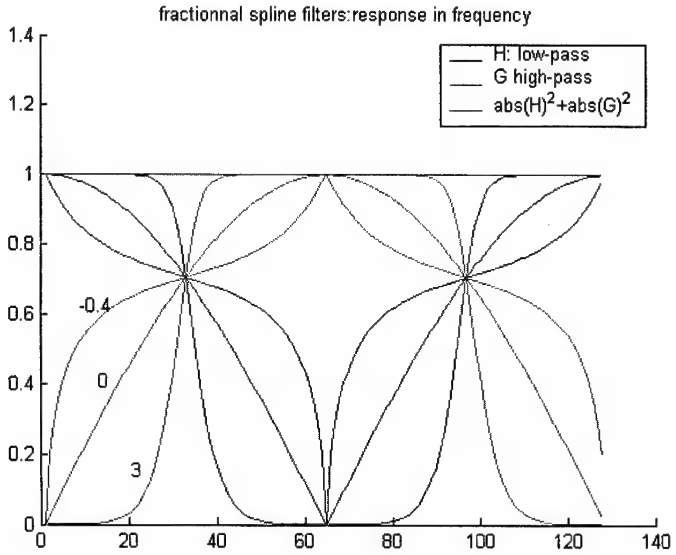
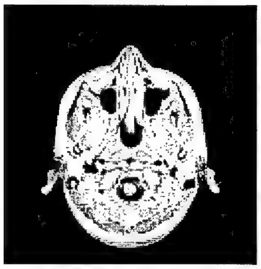
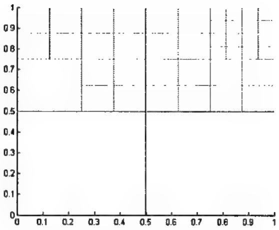
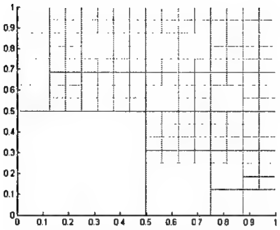
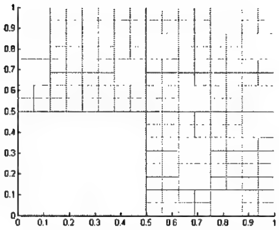
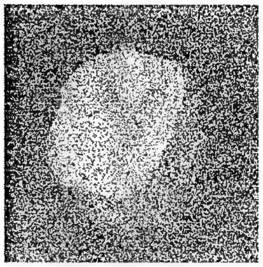
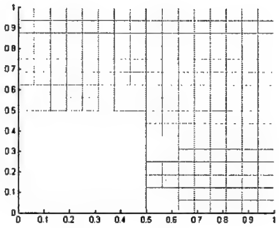
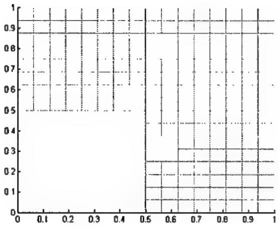
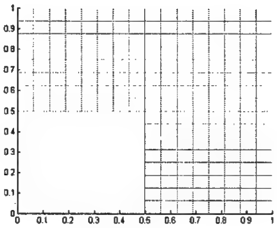
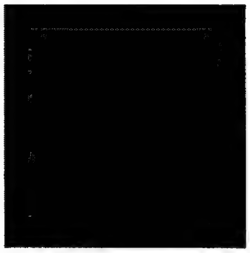
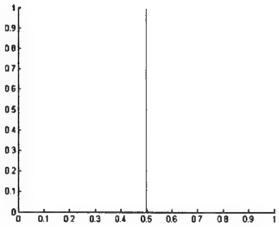
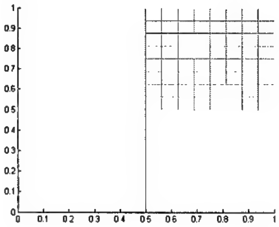
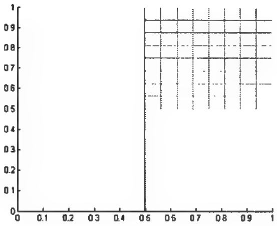
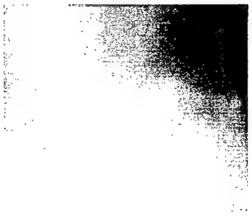
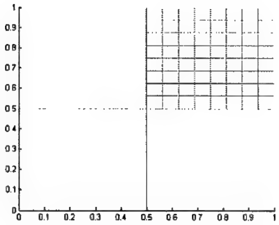
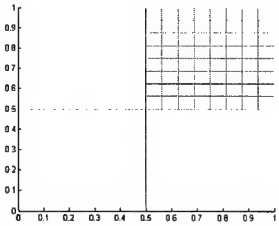
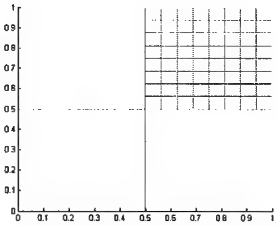
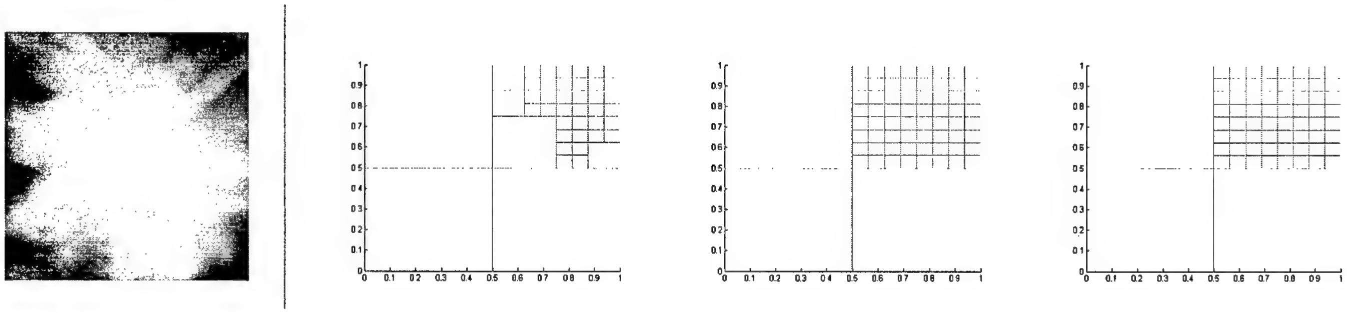
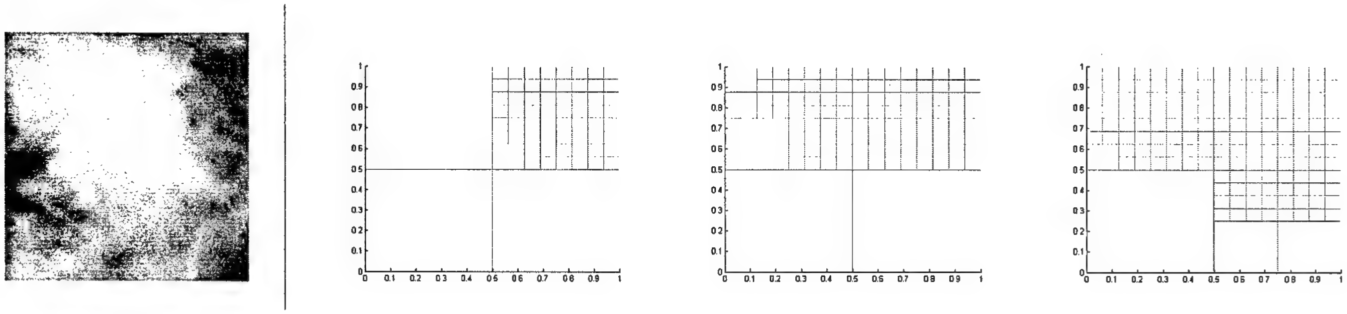


Figure 20. Selected filters.

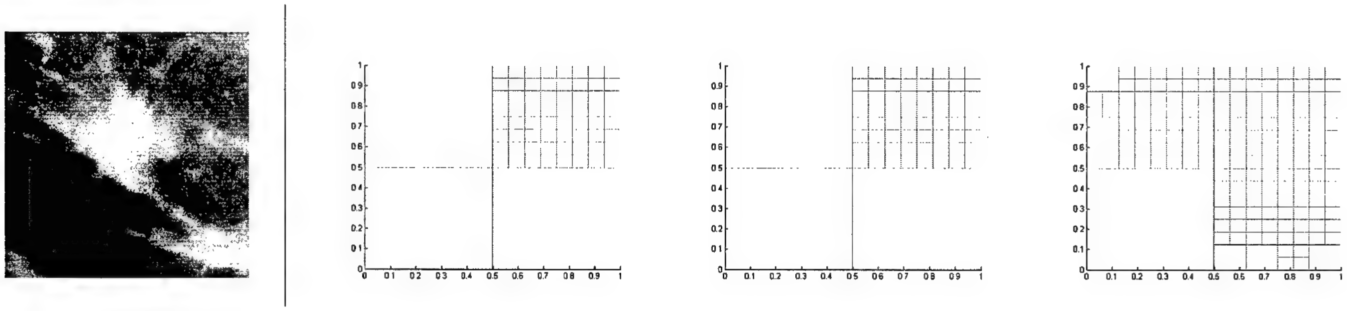
Case\alpha	-0.4	0	3
 mri			
 Mass256			
 02			
 04			


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

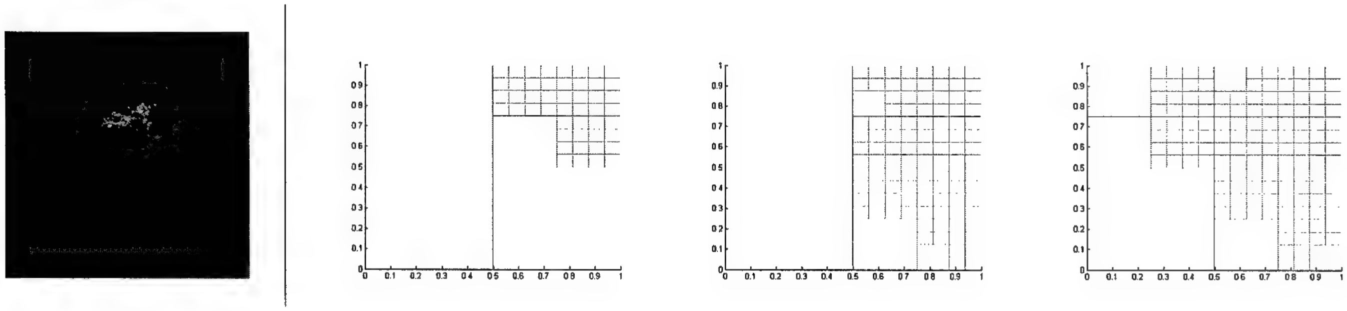
05


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

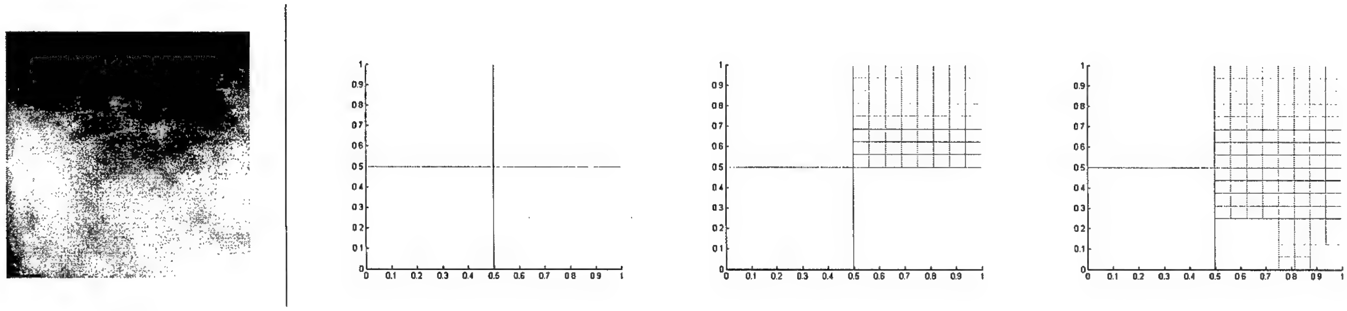
06


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

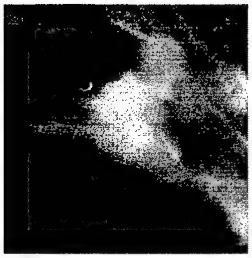
08


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

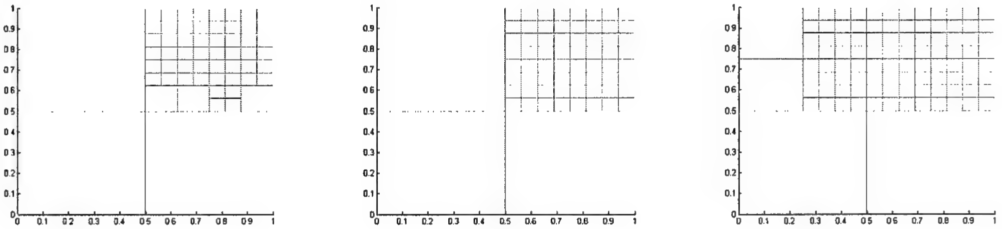
14


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

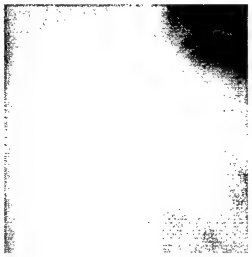
17



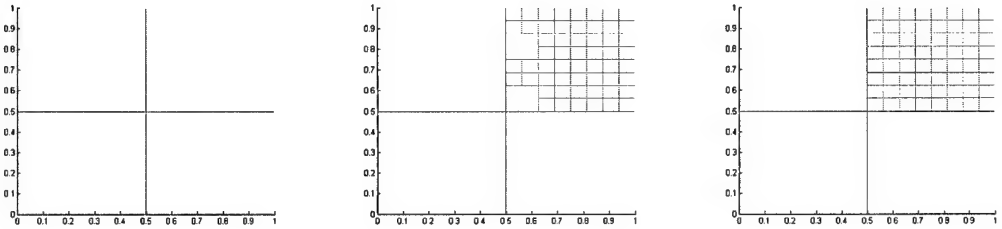
18



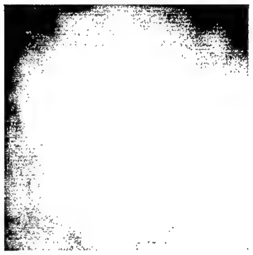
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



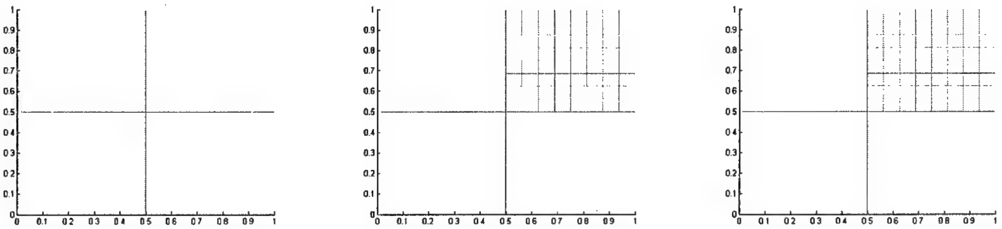
21



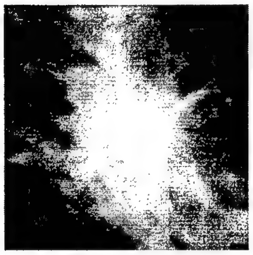
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



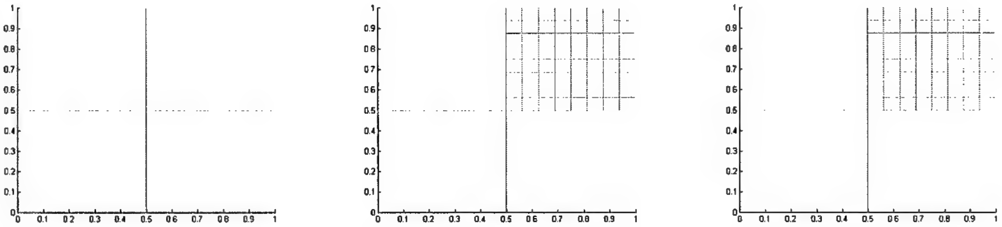
25



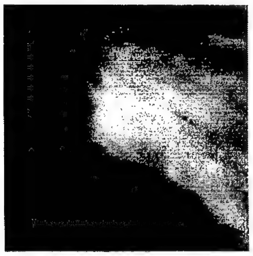
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



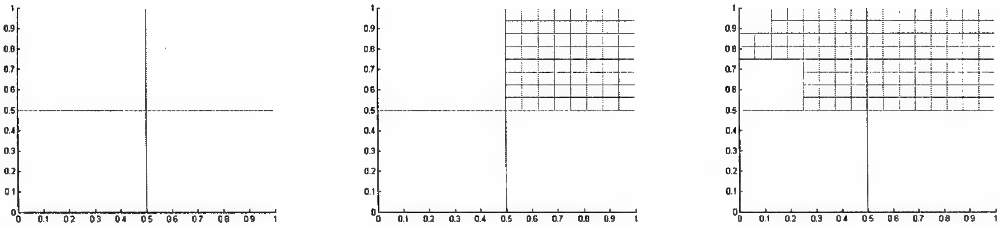
26



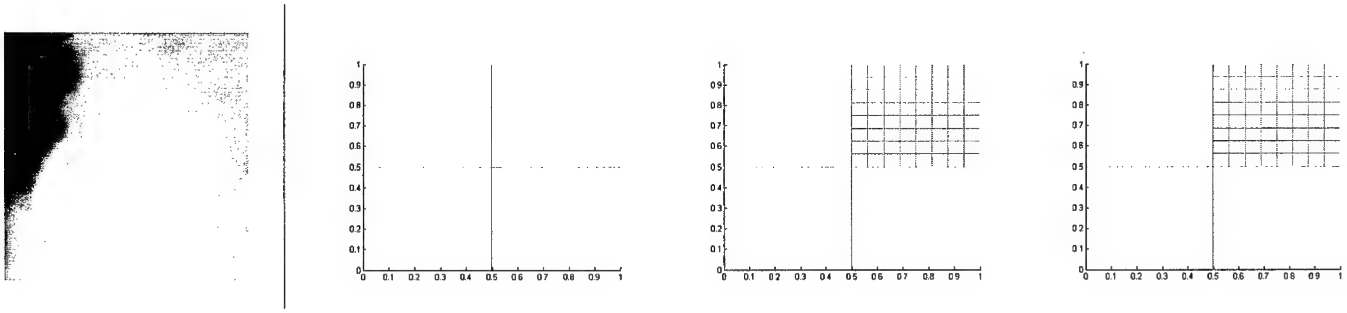
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



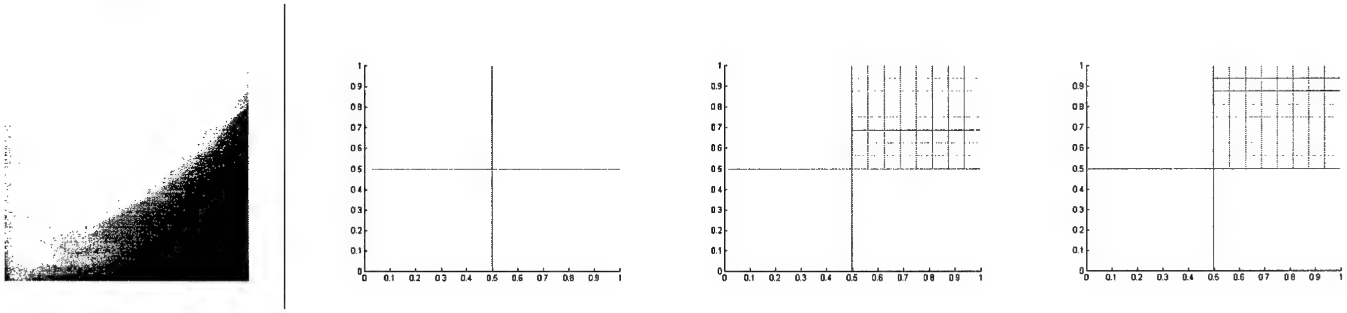
27



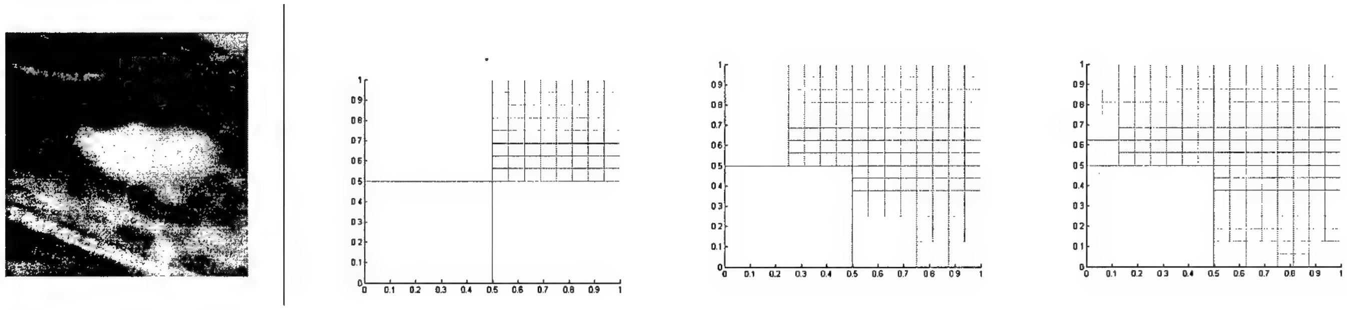
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



28



29



30

Discussion

Influence of the spline parameter

For $\alpha_1<\alpha_2$, the bandwidth of $H_{\alpha 1}$ is larger than the bandwidth of $H_{\alpha 2}$ (please see Figure 3). The frequency resolution is therefore less for $H_{\alpha 1}$ than for $H_{\alpha 2}$, but its spatial resolution is better as observed visually in Figure 8.

We also note some shifting effects perhaps due to using a non-zero phase filter - the position of the mass is changing for higher values of α (as seen for $\alpha>3$ in Figure 9).

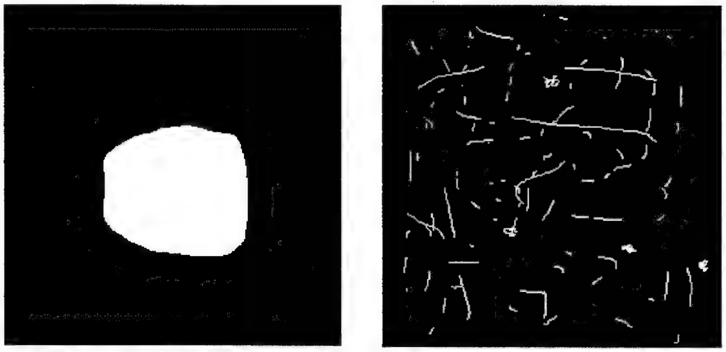
The use of fractional splines is well justified when examining Figure 9 - one can note the contrast improvement between the mass and the background, leading to a simple detection task of the mass.

For different scales and α, it is interesting to see different details. For instance, for the image shown in Figure 14, the couple (scale=4, α=-0.1) seems to extract the most visual information for the mass whereas the couple (scale=2, α=-0.5) gives a poor result in terms of detection.

A useful range of α concerning the images seems to be within the interval [-0.4,1]. And among these adjustable values of α, it would be interesting to be able to choose automatically the best one that gives high contrast between the mass and the background tissues of the breast. We did not observe a single couple (scale,α) for all our sample cases. However we identified some significant trends. We see in Figures 8 through 18 that making the value α vary within the right interval is very interesting since it allows us to see details present within the mass.

For higher values for α we observed that the details and the approximations did not result in a superior representation of the mass shown in Figure 10. Beyond these values the results are not interesting in terms of detection since the corresponding coefficients emphasized the texture within the mass.
Indeed, with a continuous analysis framework, which would allow decomposition on voices between scales, we may obtain a richer parameter space to identify the best basis for mass detection. We discuss the application of a continuous transform at the end of this section.

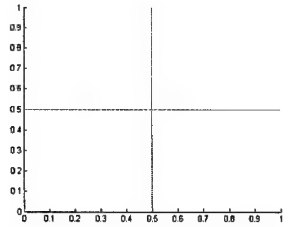
In the search for the best basis, let us first show the link between the criterion metric and detection goal. The selected coefficients within a best basis minimize the entropy of the image. However, in our case it is a these same coefficients that mostly represent the presence of a mass. Indeed, the wavelet based on the fractional splines simplifies the mass detection problem when a good choice of α is made. The minimum entropy means the maximum amount of information is contain within a minimum number of coefficients. The coefficients returned by identifying a best basis represent this information. As mention in our previous report, mammograms are largely low frequency images. Thus entropy is linked to the shape of a mass since minimizing the entropy leads to optimal correlation between basis coefficients and the mass signal (as shown in our previous year's report). To demonstrate this point, we show in Figure 21 that the entropy is lower for a signal with a mass than for an image containing high-frequency features.



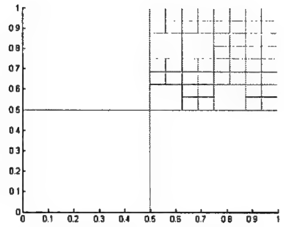
Entropy=-6.6008e+009 Entropy=-9.9922e+008

Figure 21. Samples images of different entropy.

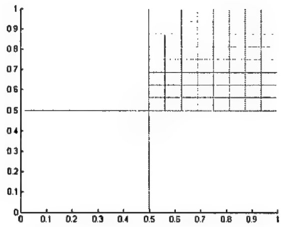
We performed experiments on three different values of the α parameter in fractional spline wavelets. The aim here was not to perform the transform for every case but to see the similarities and differences in the selected α's, and to obtain a range for computing best bases. In the next phase of the project, these bases will lead to a classification scheme for masses. At this point, we have convinced ourselves by judging the basis in terms of a visualization by reconstructing the mass from its best basis.
We observe that the expansion is often refined in the top-right quadrant, which corresponds to a first decomposition of low frequency information. In addition, we note that for higher α values, more nodes were expanded. Thus we needed to expand within coarser scales to minimize entropy.



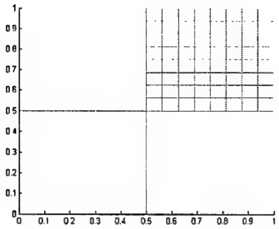
Minimum decomposition
α=-0.4
Average number of decomposed nodes: 8.2



Minimum decomposition
α=0
Average number of decomposed nodes: 23.6



Minimum decomposition
α=3
Average number of decomposed nodes: 29.4



All α 50% decomposition
Average number of decomposed nodes: 20.4

Work in progress: A Continuous Wavelet transform

During this past study of the fractional B-splines, it became clear that it would also have been useful to be able to perform a continuous transform with these functions. This has already been done in the case of classical B-spline [7]. However, we encountered problems in applying a similar scheme as in [7] in the case of fractional parameter. More specifically, we need to consider how can we apply the formula (0.14) in the case n_2 non-integer:

$$u_m^{n_2}(k) = \frac{1}{m^{n_2}} \underbrace{u_m^0 * u_m^0 \ldots * u_m^0}_{(n_2+1)\text{ times}}(k)$$

where $\{u_m^{n_2}(k)\}$ is the sequence associated to the wavelet spline functions.

It is more computationally practical to work in Fourier domain. Thus, our design approach to compute the transform for integer scales is in Fourier. We have written an m-scale relation,

$$\Phi(x/m) = \sqrt{m}\sum_{k\in\mathbb{Z}} h_{\alpha,m}(k)\Phi(x-k)$$

and we can compute

$$\frac{\hat{\beta}(m\omega)}{\hat{\beta}(\omega)} = \left(\frac{\sum_{k=0}^{m-1} e^{-jk\omega}}{m}\right)^{\alpha+1} \Rightarrow H_{\alpha,m}(e^{j\omega}) = \sqrt{\frac{A_{\varphi}^{2\alpha+1}(e^{j\omega})}{A_{\varphi}^{2\alpha+1}(e^{jm\omega})}}\sqrt{m}\left(\frac{\sum_{k=0}^{m-1} e^{-jk\omega}}{m}\right)^{\alpha+1}.$$

Now that we have the filter H, we would like to obtain the filter G that corresponds to the orthogonal wavelet filter. Our approach to this problem is illustrated in Figure 22 where h_1 and g_1 denote the new filters.

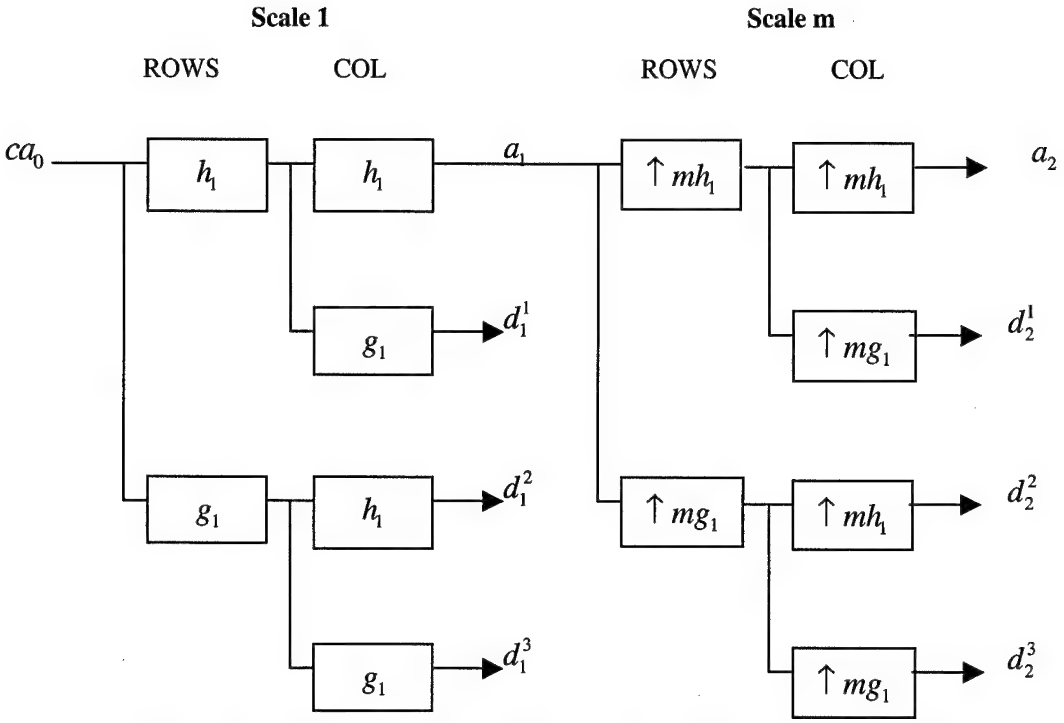


Figure 22. Filter bank for implementing a fractional B-spline wavelet expansion via a 2D CWT.

3. KEY RESEARH ACCOMPLISHMENTS

(1) During this study we first evaluated exiting wavelet functions that best matched the smoothness of a mass represented within a course level of an expansion. We then implemented under the **Matlab** programming environment an algorithm of the Continuous Discrete *fractional spline* wavelet transform in two dimensions. The algorithm was implemented without down sampling, using the "a trou Algorithm" algorithm to provide redundancy and translation invariance.

(2) We modified existing functions within the *Wavelab* library in order to adapt them to the computation of a wavelet packet representation and studied best basis search strategies using the fractional spline wavelet transform. We have also developed in interactive a tool for processing selected regions of interest.

(3) We ported this algorithm using the *Matlab* C++ compiler to generate machine code that will be integrated into our PC based Mammography Computer Aided Diagnosis (CAD) workstation for real-time interactive screening of mammography cases as described in the Statement of Work for the third year.

4. REPORTABLE OUTCOMES

The exciting results of this study will be presented at the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society on October 25, 2001, in a paper entitled "DETECTION OF MASSES IN MAMMOGRAPHY THROUGHREDUNDANT EXPANSIONS OF SCALE," V. Laffont , F. Durupt, M. A. Birgen, S. Bauduin, and A. F. Laine.

5. CONCLUSIONS

This phase of our study showed that fractional splines functions are a powerful tool for the representation of masses in mammograms and are well matched to the mass detection problem. The preliminary results presented in this report are promising but additional development is needed during Task 3. We showed that by using a continuous-varying parameter for order, that continuous analysis outperforms standard dyadic expansions. We showed that accurate approximations of mass shapes could be obtained through over complete expansions of a fractional spline wavelet transform. In the context of addressing the problem of finding the best scale (as mentioned in last year's annual report) the continuous transform implementation using wavelet packet theory appears to be a very efficient method in finding the best scale. Thus, reconstruction after thresholding of coefficients within the best basis will simplify the detection task.

The next phase of this project will focus on completing Tasks 2 and 3. We have recently taken steps to incorporate the fractional B-spline algorithm into our CAD workstation to detect masses using these representations. We will describe the performance of this approach in our next report.

REFERENCES

[1] V. Laffont, F. Durupt, M.A. Birgen, S. Bauduin, A. F. Laine "Detection of masses in mammography through redundant expansions of scale" *23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, to appear in October, 2001.

[2] M. Unser, "Splines: A Perfect Fit For Signal and Image Processing," *IEEE Signal Processing Magazine,* November 1999.

[3] M. Unser and T. Blu, "Construction of fractional spline wavelet bases," in *Proc. SPIE "Wavelets Applications in Signal and image Processing VII",* Denver, CO, 1999, vol. 3813.

[4] M. Unser and T. Blu, "Fractional splines and wavelets," *SIAM Review,* 1999.

[5] S. Mallat, "A Wavelet Tour of Signal Processing", San Diego, CA, *Academic press,*1998.

[6] M. Unser and T. Blu, "The fractional spline wavelet transform: definition and implementation", *ICASSP '00, Proceedings of the 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing,* vol. 1, 2000, pp. 512 -515, 2000.

[7] M. Unser, A. Aldroubi, S.J. Schiff, "Fast Implementation of the Continuous Wavelet Transform with Intger Scales" *IEEE Transactions on Signal Processing,* vol. 42, No. 12, December 1994.

APPENDIX

Matlab files

Bspline directory

```
function w=CDWT2Dfrac(alpha,type,scale,im)
%
% CWT2Dfracwds -- 2D Continuous Discrete Wavelet Transform
%                 using Fractional B-Splines
%   Usage
%       w=CWT2Dfracwds(alpha,type,scale,im)
%   Inputs
%       alpha       spline parameter alpha>-0.5
%       type        see FFTfractsplinefilters
%       scale       scale at which the transform will be computed
%       im          image 2D
%   Outputs
%       w           coefficients of the transform
%
%
%   See Also:
%   FFTfractsplinefilters, simu_alpha_scale



S=size(im);
n=S(1);
nc=S(2);


if n~=nc
    disp('ERROR : image must be square!')
    break
end
J1=log2(n);
J2=ceil(J1);
if J1~=J2
    disp('ERROR : the size of the input image must be a power of two!')
    break
end

imfft=fft2(im);
m=n;
w=[];
f=[1:m];
q=1;
ag=imfft;

for k=1:scale

    [H1,H2]=FFTfractsplinefilters(m,alpha,type,q);
    G=H1(1,:);
    G=1/sqrt(2).*G;
    H=H1(2,:);
    H=1/sqrt(2).*H;
    %computation on the rows
    for i=1:m
        %low-pass filtering
        agl(i,:)=ag(i,:).*G;
        %high-pass filtering
        ahl(i,:)=ag(i,:).*H;
    end
    %computation on the columns
    for j=1:m
        a(:,j)=agl(:,j).*G.';
        d1(:,j)=agl(:,j).*H.';
        d2(:,j)=ahl(:,j).*G.';
        d3(:,j)=ahl(:,j).*H.';
    end
    w1=ifft2(d1);
    w2=ifft2(d2);
    w3=ifft2(d3);
    w1=w1+min(min(w1));
    w2=w2+min(min(w2));
    w3=w3+min(min(w3));

    w=[w1 w2 w3 w];
    ag=a;

    q=q*2;

end
approx=ifft2(a);
approx=approx+min(min(approx));


w=([w approx]);
```

```
function
[FFTanalysisfilters,FFTsynthesisfilters]=FFTfractsplinefilters(M,alpha,
type,q)
% FFTFRACSPLINEFILTERS Filters for the various orthogonal fractional
%      spline wavelet transforms (orthonormal or semi-orthonormal).
%
      [FFTanalysisfilters,FFTsynthesisfilters]=FFTfractsplinefilters(M,
alpha,type)
%      Computation of the frequency response of the low- and high-pass
%      filters that generate the orthonormal or semi-orthonormal (B-
spline
%      or dual) fractional splines of degree alpha, and of type '+'
(causal),
%       '-' (anticausal) or '*' (symmetric).
%
%      Input:
%      M : size of the input signal = length of FFTfilters = 2^N
%      alpha>-0.5 : degree of the fractional splines
%      type  : type of B-splines
%            = '+ortho' (causal orthonormal, default), '-ortho'
(anticausal
%            orthonormal) or '*ortho' (symmetric orthonormal);
%            = '+bspline' (causal B-spline), '-bspline' (anticausal
%            B-spline) or '*bspline' (symmetric B-spline).
%            = '+dual' (causal dual), '-dual' (anticausal dual) or
'*dual'
%            (symmetric dual). The last option is the flipped version of
the
%            B-spline one.
%    q : relative to the scale: the filters change from a scale to the
other
%      Output:
%      FFTanalysisfilters      = [lowpassfilter;highpassfilter]    : FFT
filter arrays
%      FFTsynthesisfilters     = [lowpassfilter;highpassfilter]    : FFT
filter arrays
%
%      See also FFTWAVELETANALYSIS, FFTWAVELETSYNTHESIS.
%      Uses FRACTSPLINEAUTOCORR
%
%      Author: Thierry Blu, October 1999
%      Biomedical Imaging Group, EPFL, Lausanne, Switzerland.
%      This software is downloadable at http://bigwww.epfl.ch/
%
%      References:
%      [1] M. Unser and T. Blu, "Fractional splines and wavelets,"
%      SIAM Review, Vol. 42, No. 1, pp. 43--67, January 2000.
%      [2] M. Unser and T. Blu, "Construction of fractional spline
wavelet bases,"
%      Proc. SPIE, Wavelet Applications in Signal and Image Processing
VII,
%      Denver, CO, USA, 19-23 July, 1999, vol. 3813, pp. 422-431.
%      [3] T. Blu and M. Unser, "The fractional spline wavelet
transform: definition and
%      implementation," Proc. IEEE International Conference on
Acoustics, Speech, and
```

```
%     Signal Processing (ICASSP'2000), Istanbul, Turkey, 5-9 June 2000,
vol. I, pp. 512-515 .

if alpha<=-0.5
      disp('The autocorrelation of the fractional splines exists only
')
      disp('for degrees strictly larger than -0.5!')
      FFTanalysisfilters=[];
      FFTsynthesisfilters=[];
      return
end
if M~=2^round(log(M)/log(2))
      disp(' ')
      disp('The size of the FFT must be a power of two!')
      disp(' ')
      FFTanalysisfilters=[];
      FFTsynthesisfilters=[];
      return
end
nu=0:1/M:(1-1/M);

if nargin<3
      type='+ortho';
end
if length(type)<=1
      error(['''' type '''' ' is an unknown filter type!'])
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%NEW%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A=fractsplineautocorr(alpha,nu*q);
A2=fractsplineautocorr(alpha,nu*(2*q));
%A2=[A A];'
%A2=A2(1:2:length(A2));       % A2(z) = A(z^2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nu=nu*q;%%%%%%%%%%NEW%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if type(2)=='o'|type(2)=='O'
      % orthonormal spline filters
      if type(1)=='*'
            lowa=sqrt(2)*abs((1+exp(-
2*i*pi*nu))/2).^(alpha+1).*sqrt(A./A2);
      else
            if type(1)=='+'
                  lowa=sqrt(2)*((1+exp(-
2*i*pi*nu))/2).^(alpha+1).*sqrt(A./A2);
            else
                  if type(1)=='-'

      lowa=sqrt(2)*((1+exp(2*i*pi*nu))/2).^(alpha+1).*sqrt(A./A2);
                  else
                        error(['''' type(1) '''' ' is an unknown filter
prefix!'])
                  end
            end
      end
      %higha=exp(-2*i*pi*nu).*lowa;
      %higha=conj([higha(M/2+(1:M/2)) higha(1:M/2)]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
      higha=exp(-2*i*pi*nu).*lowa;
      higha=conj([higha(M/(2*q)+(1:M-M/(2*q))) higha(1:M/(2*q))]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
      lows=lowa;
      highs=higha;
      FFTanalysisfilters=[lowa;higha];
      FFTsynthesisfilters=[lows;highs];
else
      % semi-orthonormal spline filters
      if type(1)=='*'
            lowa=sqrt(2)*abs((1+exp(-2*i*pi*nu))/2).^(alpha+1);
      else
            if type(1)=='+'
                  lowa=sqrt(2)*((1+exp(-2*i*pi*nu))/2).^(alpha+1);
            else
                  if type(1)=='-'
                        lowa=sqrt(2)*((1+exp(2*i*pi*nu))/2).^(alpha+1);
                  else
                        error(['''' type '''' ' is an unknown filter
type!'])
                  end
            end
      end
      higha=exp(-2*i*pi*nu).*lowa.*A;
      higha=conj([higha(M/2+(1:M/2)) higha(1:M/2)]);

      lows=lowa.*A./A2;
      highs=higha./(A2.*[A(M/2+(1:M/2)) A(1:M/2)]);
      if type(2)=='d'|type(2)=='D'
            % dual spline wavelets
            FFTanalysisfilters=[lowa;higha];
            FFTsynthesisfilters=[lows;highs];
      else
            % B-spline wavelets
            if type(2)=='b'|type(2)=='B'
                  FFTsynthesisfilters=[lowa;higha];
                  FFTanalysisfilters=[lows;highs];
            else
                  error(['''' type '''' ' is an unknown filter type!'])
            end
      end
end
```

```
function A=fractsplineautocorr(alpha,nu)

% FRACTSPLINEAUTOCORR Frequency domain computation of fractional spline
%     autocorrelation.  A=fractsplineautocorr(alpha,nu) computes the
%     frequency response of the autocorrelation filter
A(exp(2*i*Pi*nu))
%     of a fractional spline of degree alpha.  It uses an acceleration
%     technique to improve the convergence of the infinite sum by 4
%     orders.
%
%     See also FFTSPLINEFILTERS
%
%     Author: Thierry Blu, October 1999
%     Biomedical Imaging Group, EPFL, Lausanne, Switzerland.
%     This software is downloadable at http://bigwww.epfl.ch/
%
%     References:
%     [1] M. Unser and T. Blu, "Fractional splines and wavelets,"
%     SIAM Review, Vol. 42, No. 1, pp. 43--67, January 2000.
%     [2] M. Unser and T. Blu, "Construction of fractional spline
wavelet bases,"
%     Proc. SPIE, Wavelet Applications in Signal and Image Processing
VII,
%     Denver, CO, USA, 19-23 July, 1999, vol. 3813, pp. 422-431.
%     [3] T. Blu and M. Unser, "The fractional spline wavelet
transform: definition and
%     implementation," Proc. IEEE International Conference on
Acoustics, Speech, and
%     Signal Processing (ICASSP'2000), Istanbul, Turkey, 5-9 June 2000,
vol. I, pp. 512-515 .

N=100;                  % number of terms of the summation for
computing
                        % the autocorrelation frequency response

if alpha<=-0.5
      disp('The autocorrelation of the fractional splines exists only
')
      disp('for degrees strictly larger than -0.5!')
      A=[];
      return
end

S=zeros(1,length(nu));
err=[];
err0=[];
for n=-N:N
      S=S+abs(sinc(nu+n)).^(2*alpha+2);
end
U=2/(2*alpha+1)/N^(2*alpha+1);
U=U-1/N^(2*alpha+2);
U=U+(alpha+1)*(1/3+2*nu.*nu)/N^(2*alpha+3);
U=U-(alpha+1)*(2*alpha+3)*nu.*nu/N^(2*alpha+4);
U=U.*abs(sin(pi*nu)/pi).^(2*alpha+2);
A=S+U;
```

```
function A=fractsplineautocorr(alpha,nu)

% FRACTSPLINEAUTOCORR Frequency domain computation of fractional spline
%      autocorrelation.  A=fractsplineautocorr(alpha,nu) computes the
%      frequency response of the autocorrelation filter
A(exp(2*i*Pi*nu))
%      of a fractional spline of degree alpha.  It uses an acceleration
%      technique to improve the convergence of the infinite sum by 4
%      orders.
%
%      See also FFTSPLINEFILTERS
%
%      Author: Thierry Blu, October 1999
%      Biomedical Imaging Group, EPFL, Lausanne, Switzerland.
%      This software is downloadable at http://bigwww.epfl.ch/
%
%      References:
%      [1] M. Unser and T. Blu, "Fractional splines and wavelets,"
%      SIAM Review, Vol. 42, No. 1, pp. 43--67, January 2000.
%      [2] M. Unser and T. Blu, "Construction of fractional spline
wavelet bases,"
%      Proc. SPIE, Wavelet Applications in Signal and Image Processing
VII,
%      Denver, CO, USA, 19-23 July, 1999, vol. 3813, pp. 422-431.
%      [3] T. Blu and M. Unser, "The fractional spline wavelet
transform: definition and
%      implementation," Proc. IEEE International Conference on
Acoustics, Speech, and
%      Signal Processing (ICASSP'2000), Istanbul, Turkey, 5-9 June 2000,
vol. I, pp. 512-515 .

N=100;                  % number of terms of the summation for
computing
                        % the autocorrelation frequency response

if alpha<=-0.5
      disp('The autocorrelation of the fractional splines exists only
')
      disp('for degrees strictly larger than -0.5!')
      A=[];
      return
end

S=zeros(1,length(nu));
err=[];
err0=[];
for n=-N:N
      S=S+abs(sinc(nu+n)).^(2*alpha+2);
end
U=2/(2*alpha+1)/N^(2*alpha+1);
U=U-1/N^(2*alpha+2);
U=U+(alpha+1)*(1/3+2*nu.*nu)/N^(2*alpha+3);
U=U-(alpha+1)*(2*alpha+3)*nu.*nu/N^(2*alpha+4);
U=U.*abs(sin(pi*nu)/pi).^(2*alpha+2);
A=S+U;
```

Best basis directory

```
function [bb,value]=test_best(im,MaxDeep,alpha)
% test_best -- main program for best basis reasearch
%  Usage
%    [bb,value] = test_best(im,MaxDeep,alpha)
%  Inputs
%    im        image 2D
%    MaxDeep   maximum depth of tree-search
%    alpha     spline parameter
%  Outputs
%    bb        basis-quadtree of best basis
%    value     value of components of best basis
%
%
%
%  See Also
%  sb_Calc2dStatTree, sb_Best2dBasis, sb_coef_basis, sb_Plot2dPartition

name_dir=['output_alpha',num2str(alpha,3),'_deep',num2str(MaxDeep,3)];
mkdir(name_dir);

cd (name_dir)

im=fft2(im);

[stats] =  sb_Calc2dStatTree(im,MaxDeep,alpha,'Entropy',0.2);

[bb value]    =  sb_Best2dBasis(stats,MaxDeep);
bb(1)=1;
bb=sb_coef_basis(MaxDeep,bb);
save best_basis.mat bb
bb1=find(bb);
disp(['decomposed nodes = ',num2str(length(bb1))])

cd ..

figure
title(['alpha= ',num2str(alpha,3),' MaxDeep= ',num2str(MaxDeep,3)])
ax=axis;hold on
sb_Plot2dPartition(bb,ax,'y')
```

```
function [SQTree] = sb_Calc2dStatTree(img,D,alpha,ent,EntPar)%TFPar,
% sb_Calc2dStatTree -- Put Wavelet/ Statistics into Quad Tree
%  Usage
%    SQTree = Calc2dStatTree(img,D,alpha,ent[,EntPar])
%  Inputs
%    img       2-d image, n by n, n dyadic
%    D         maximum depth of splitting
%    alpha     spline parameter
%    ent       type of entropy to record in tree: options are
%                 'Entropy' -- Coifman-Wickerhauser ==> We use this one
%                 'Log'     -- sum log(abs(th_i))
%                 '1^p'     -- sum |th_i|^p, 0 < p < 2, p = EntPar
%                 'N(eps)'  -- #>= eps, eps = EntPar
%                 'Risk'    -- sum min(th_i^2,eps^2), eps = EntPar
%                 'Sum'     -- sum th_i
%                 'SURE'    -- SURE(Thresholding), thresh = EntPar
%    EntPar    extra parameter, depends on type of entropy
%  Outputs
%    sqtree    quad tree filled entropy numbers.
%              SQTree(qnode(d,bx,by)) contains entropy of
quadlet(d,bx,by,n)
%
%  Description
%    A packet table is indexed by depth, block within depth, and
%    coefficient within block. 2d images require four
%    splits per parent block, we prefers to store the
%    entropy measure for the coefficients in a particular block
%    and the coefficients themselves.  sb_Calc2dStatTree
%    creates this statistics tree for the entropy measure chosen
%    by the user.
%    !!!  the stored coefficients are in the Fourier domain !!!
%  See Also
%    sb_Best2dBasis, sb_Plot2dBasisTree
%


        [n1,n2]=size(img);
        boxcnt = 1;
        SQTree = zeros(1, 2* (4^D));
        name=['coef_0_0_0.dat'];
        save_volume(img,name,'float32');
        if nargin == 4, EntPar = []; end


          for deep = 0:D,

[qmf,FFTsynthesisfilters]=FFTfractsplinefilters(n1,alpha,'+ortho',2^(de
ep));
              qmf=1/sqrt(2).*qmf;
            for bx=0:(boxcnt-1),
              for by=0:(boxcnt-1),


name=['coef_',num2str(deep),'_',num2str(bx),'_',num2str(by),'.dat'];
                A=read_volume(name,[n1 n2 1 1],'float32');

                object=ifft2(A);
                ss = norm(norm(object));
                object = object ./(ss);
                SQTree(qnode(deep,[bxby])) =
sb_CalcEntropy(object,'Entropy',EntPar);
                % We divide at each scale the enropy by 4 in order to
ensure the normalization of the entropy
                SQTree(qnode(deep,[bx by])) = SQTree(qnode(deep,[bx
by]))/4^(deep);
                disp(['d bx by =
',num2str(deep),num2str(bx),num2str(by)]);

                if deep < D, %        prepare for finer scales
                  [q1,q2,q3,q4] = sb_DownQuad(A,qmf);


name1=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by),'.da
t'];

name2=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by),'.
dat'];

name3=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by+1),'.
dat'];

name4=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by+1),
'.dat'];
                  save_volume(q1,name1,'float32');
                  save_volume(q2,name2,'float32');
                  save_volume(q3,name3,'float32');
                  save_volume(q4,name4,'float32');
                  clear q1 q2 q3 q4
                end
              end
          end
            boxcnt = boxcnt*2;

      end
```

```
function Ent = sb_CalcEntropy(object,Entropy,par)
% CalcEntropy -- Calculate entropy number of array
%  Usage
%    Ent = CalcEntropy(object,ent[,par])
%  Inputs
%    object    1-d or 2-d object
%    ent       type of entropy to use: options are
%                 'Entropy' -- Coifman-Wickerhauser
%                 'Log'     -- sum log(|th_i|)
%                 'l^p'     -- sum |th_i|^p, 0 < p < 2, p = par
%                 'N(eps)'  -- #>= eps, eps = par
%                 'Risk'    -- sum min(th_i^2,eps^2), eps=par
%                 'Sum'     -- sum th_i
%                 'SURE'    -- SURE(Thresholding), thresh = par
%    par       extra parameter, depends on type of entropy
%  Outputs
%    Ent       Entropy of object
%
%  Description
%    It is traditional to normalize the object to norm 1 before
%    calling this routine.  This routine does NO pre-scaling.
%
%  See Also
%    sb_Calc2dStatTree
%


      if strcmp(Entropy,'Entropy'),
            p = (object(:) ).^2;
            Ent = - sum(sum(p .* log(p)));
      elseif strcmp(Entropy,'Log')
            p = abs(object(:) );
            Ent = sum(log(eps+p));
      elseif strcmp(Entropy,'l^p'),
            % par = p = exponent
            p = abs(object(:) );
            Ent = sum(p .^par);
      elseif strcmp(Entropy,'N(eps)'),
            % par = eps
            p = abs(object(:));
            Ent = sum( p > par );
      elseif strcmp(Entropy,'Risk'),
            p = object(:).^2;
            Ent = sum(min(p,par^2));
      elseif strcmp(Entropy,'Sum'),
            p = object(:);
            Ent = sum(p);
      elseif strcmp(Entropy,'SURE'),
            tt = par^2;  % par = threshold
            dim = length(object(:));
            p = object(:).^2;
            ngt = sum( p > tt);
            nlt = dim - ngt;
            slt = sum( p .* ( p <= tt ));
            Ent = dim - 2*nlt + tt*ngt + slt;
      else
            disp('in CalcEntropy: unknown Entropy request')
            fprintf('Request=<<%s>>\n',Entropy)
            disp('I only know how to do:')
            disp(' Entropy, Log, l^p, N(eps), Risk, SURE ')
      end
clear object


function bb=sb_coef_basis(D,basis)
% sb_coef_basis -- return the best basis tree
%  Usage
%    bb=sb_coef_basis(D,basis)
%  Inputs
%    basis   best basis reurned by sb_Best2DBasis
%    D       Maximum depth of decomposition
%  Outputs
%    bb      best basis
%
%  Description
%    sb_Best2dBasis returns a tree with 1 and 0
%    but it can have 1 on a depth and only 0 on
%    the depth before...
%    the coefficients that are non terminal nodes
%    are deleted

b=[];

for deep = 0:D-1,
      for bx=0:(2^deep-1),
            for by=0:(2^deep-1),

                    if(basis(qnode(deep,[bx,by])) == 0) ,  %
nonterminal node
                        basis(qnode(deep+1,[2*bx,2*by])) = 0;
                        basis(qnode(deep+1,[2*bx+1,2*by])) = 0;
                        basis(qnode(deep+1,[2*bx,2*by+1])) = 0;
                        basis(qnode(deep+1,[2*bx+1,2*by+1])) = 0;
                    end
            end
       end
 end

 bb=basis;
 find(basis);
 for deep = D-1:-1:0,
      for bx=0:(2^deep-1),
            for by=0:(2^deep-1),
                    if(basis(qnode(deep,[bx,by])) == 0) ,  %
nonterminal node


name=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by),'.dat
'];
                        delete (name)

name=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by),'.d
at'];
                        delete (name)
```

```
name=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by+1),'.d
at'];
                        delete (name)

name=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by+1),'
.dat'];
                        delete (name)
                    else

                        if (basis(qnode(deep+1,[2*bx,2*by])) == 1)

name=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by),'.dat
'];
                            delete (name)
                        end
                        if (basis(qnode(deep+1,[2*bx+1,2*by])) == 1)

name=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by),'.d
at'];
                            delete (name)
                        end
                        if (basis(qnode(deep+1,[2*bx,2*by+1])) == 1)

name=['coef_',num2str(deep+1),'_',num2str(2*bx),'_',num2str(2*by+1),'.d
at'];
                            delete (name)
                        end
                        if (basis(qnode(deep+1,[2*bx+1,2*by+1])) == 1)

name=['coef_',num2str(deep+1),'_',num2str(2*bx+1),'_',num2str(2*by+1),'
.dat'];
                            delete (name)
                        end
                    end
            end
        end
    end

    delete('coef_0_0_0.dat')
```